



Red Hat Satellite 6.14

API Guide

Develop custom applications or integrations by using the Satellite REST API

Red Hat Satellite 6.14 API Guide

Develop custom applications or integrations by using the Satellite REST API

Red Hat Satellite Documentation Team

satellite-doc-list@redhat.com

Legal Notice

Copyright © 2025 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

The Red Hat Satellite Representational State Transfer (REST) API guide explains the concepts behind a REST API and provides example usage for various types of requests. This provides a basis for administrators and developers to write custom scripts and integrate Red Hat Satellite with third-party applications.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	3
CHAPTER 1. INTRODUCTION	4
1.1. OVERVIEW OF THE RED HAT SATELLITE API	4
1.2. SATELLITE API COMPARED TO HAMMER CLI TOOL	4
CHAPTER 2. API REFERENCE	5
2.1. UNDERSTANDING THE API SYNTAX	5
2.1.1. Using the GET HTTP verb	6
2.1.2. Using the POST HTTP verb	6
2.1.3. Using the PUT HTTP verb	7
2.1.4. Using the DELETE HTTP verb	8
2.1.5. Relating API error messages to the API reference	9
2.2. UNDERSTANDING THE JSON RESPONSE FORMAT	9
CHAPTER 3. AUTHENTICATING API CALLS	12
3.1. SSL AUTHENTICATION OVERVIEW	12
3.1.1. Configuring SSL authentication	12
3.2. HTTP AUTHENTICATION OVERVIEW	12
3.3. PERSONAL ACCESS TOKEN AUTHENTICATION OVERVIEW	13
3.3.1. Creating a Personal Access Token	13
3.3.2. Revoking a Personal Access Token	14
3.4. OAUTH AUTHENTICATION OVERVIEW	14
3.4.1. Configuring OAuth	14
3.4.2. OAuth request format	15
CHAPTER 4. API REQUESTS IN DIFFERENT LANGUAGES	16
4.1. API REQUESTS WITH CURL	16
4.1.1. Passing JSON data to the API request	16
4.1.2. Retrieving a list of resources	17
4.1.3. Creating and modifying resources	19
4.2. API REQUESTS WITH RUBY	19
4.2.1. Creating objects using Ruby	19
4.2.2. Using apipie bindings with Ruby	21
4.3. API REQUESTS WITH PYTHON	22
4.3.1. Creating objects using Python	22
4.3.2. Requesting information from the API using Python	25
CHAPTER 5. USING THE RED HAT SATELLITE API	28
5.1. WORKING WITH HOSTS	28
5.2. WORKING WITH LIFE CYCLE ENVIRONMENTS	31
5.3. UPLOADING CONTENT TO THE SATELLITE SERVER	35
5.4. APPLYING ERRATA TO A HOST OR HOST COLLECTION	39
5.5. USING EXTENDED SEARCHES	40
5.6. USING SEARCHES WITH PAGINATION CONTROL	41
5.7. OVERRIDING SMART CLASS PARAMETERS	42
5.8. MODIFYING A SMART CLASS PARAMETER USING AN EXTERNAL FILE	43
5.9. DELETING OPENSAP REPORTS	46
5.10. WORKING WITH PULP USING SATELLITE API	48
APPENDIX A. API RESPONSE CODES	51
APPENDIX B. API PERMISSIONS MATRIX	52

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Use the **Create Issue** form in Red Hat Jira to provide your feedback. The Jira issue is created in the Red Hat Satellite Jira project, where you can track its progress.

Prerequisites

- Ensure you have registered a [Red Hat account](#).

Procedure

1. Click the following link: [Create Issue](#). If Jira displays a login error, log in and proceed after you are redirected to the form.
2. Complete the **Summary** and **Description** fields. In the **Description** field, include the documentation URL, chapter or section number, and a detailed description of the issue. Do not modify any other fields in the form.
3. Click **Create**.

CHAPTER 1. INTRODUCTION

Red Hat Satellite provides a Representational State Transfer (REST) API. The API provides software developers and system administrators with control over their Red Hat Satellite environment outside of the standard web interface. The REST API is useful for developers and administrators who aim to integrate the functionality of Red Hat Satellite with custom scripts or external applications that access the API over HTTP.

1.1. OVERVIEW OF THE RED HAT SATELLITE API

The benefits of using the REST API are:

- Broad client support – any programming language, framework, or system with support for HTTP protocol can use the API.
- Self-descriptive – client applications require minimal knowledge of the Red Hat Satellite infrastructure because a user discovers many details at runtime.
- Resource-based model – the resource-based REST model provides a natural way to manage a virtualization platform.

You can use the REST API to perform the following tasks:

- Integrate with enterprise IT systems.
- Integrate with third-party applications.
- Perform automated maintenance or error checking tasks.
- Automate repetitive tasks with scripts.

As you prepare to upgrade Satellite Server, ensure that any scripts you use that contain Satellite API commands are up to date. API commands differ between versions of Satellite.

1.2. SATELLITE API COMPARED TO HAMMER CLI TOOL

For many tasks, you can use both Hammer and Satellite API. You can use Hammer as a human-friendly interface to Satellite API. For example, to test responses to API calls before applying them in a script, use the **--debug** option to inspect API calls that Hammer issues: **hammer --debug organization list**.

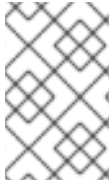
In the background, each Hammer command first establishes a binding to the API and then sends a request. This can have performance implications when executing a large number of Hammer commands in sequence. In contrast, scripts that use API commands communicate directly with the Satellite API.

Note that you must manually update scripts that use API commands, while Hammer automatically reflects changes in the API. For more information, see the [Hammer CLI Guide](#).

CHAPTER 2. API REFERENCE

The full API reference is available on your Satellite Server at <https://satellite.example.com/apidoc/v2.html>. Be aware that even though versions 1 and 2 of the Satellite 6 API are available, Red Hat only supports version 2.

2.1. UNDERSTANDING THE API SYNTAX



NOTE

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

The built-in API reference shows the API route, or path, preceded by an HTTP verb:

HTTP_VERB API_ROUTE

To work with the API, construct a command using the **curl** command syntax and the API route from the reference document:

```
$ curl --request HTTP_VERB \
--insecure \
--user sat_username:sat_password \
--data @file.json \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--output file
API_ROUTE \
| python3 -m json.tool
```

- 1 To use **curl** for the API call, specify an HTTP verb with the **--request** option. For example, **--request POST**.
- 2 Add the **--insecure** option to skip SSL peer certificate verification check.
- 3 Provide user credentials with the **--user** option.
- 4 For **POST** and **PUT** requests, use the **--data** option to pass JSON formatted data. For more information, see [Section 4.1.1, "Passing JSON data to the API request"](#).
- 5 6 When passing the JSON data with the **--data** option, you must specify the following headers with the **--header** option. For more information, see [Section 4.1.1, "Passing JSON data to the API request"](#).
- 7 When downloading content from Satellite Server, specify the output file with the **--output** option.
- 8 Use the API route in the following format:
https://satellite.example.com/katello/api/activation_keys. In Satellite 6, version 2 of the API is the default. Therefore it is not necessary to use **v2** in the URL for API calls.
- 9 Redirect the output to the Python **json.tool** module to make the output easier to read.

2.1.1. Using the GET HTTP verb

Use the GET HTTP verb to get data from the API about an existing entry or resource.

Example

This example requests the amount of Satellite hosts:

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/hosts | python3 -m json.tool
```

Example response:

```
{
  "total": 2,
  "subtotal": 2,
  "page": 1,
  "per_page": 20,
  "search": null,
  "sort": {
    "by": null,
    "order": null
  },
  "results":
    output truncated
```

The response from the API indicates that there are two results in total, this is the first page of the results, and the maximum results per page is set to 20. For more information, see [Section 2.2, “Understanding the JSON response format”](#).

2.1.2. Using the POST HTTP verb

Use the POST HTTP verb to submit data to the API to create an entry or resource. You must submit the data in JSON format. For more information, see [Section 4.1.1, “Passing JSON data to the API request”](#).

Example

This example creates an activation key.

1. Create a test file, for example, **activation-key.json**, with the following content:

```
{ "organization_id": 1, "name": "TestKey", "description": "Just for testing" }
```

2. Create an activation key by applying the data in the **activation-key.json** file:

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request POST \
--user sat_username:sat_password --insecure \
--data @activation-key.json \
https://satellite.example.com/katello/api/activation_keys \
| python3 -m json.tool
```

Example response:

```
{
  "id": 2,
  "name": "TestKey",
  "description": "Just for testing",
  "unlimited_hosts": true,
  "auto_attach": true,
  "content_view_id": null,
  "environment_id": null,
  "usage_count": 0,
  "user_id": 3,
  "max_hosts": null,
  "release_version": null,
  "service_level": null,
  "content_overrides": [

  ],
  "organization": {
    "name": "Default Organization",
    "label": "Default_Organization",
    "id": 1
  },
  "created_at": "2017-02-16 12:37:47 UTC",
  "updated_at": "2017-02-16 12:37:48 UTC",
  "content_view": null,
  "environment": null,
  "products": null,
  "host_collections": [

  ],
  "permissions": {
    "view_activation_keys": true,
    "edit_activation_keys": true,
    "destroy_activation_keys": true
  }
}
```

3. Verify that the new activation key is present. In the Satellite web UI, navigate to **Content** > **Activation keys** to view your activation keys.

2.1.3. Using the PUT HTTP verb

Use the PUT HTTP verb to change an existing value or append to an existing resource. You must submit the data in JSON format. For more information, see [Section 4.1.1, "Passing JSON data to the API request"](#).

Example

This example updates the **TestKey** activation key created in the previous example.

1. Edit the **activation-key.json** file created previously as follows:

```
{"organization_id":1, "name":"TestKey", "description":"Just for testing", "max_hosts":"10" }
```

2. Apply the changes in the JSON file:

Example request:

```
$ curl --header "Accept:application/json" \  
--header "Content-Type:application/json" --request PUT \  
--user sat_username:sat_password --insecure \  
--data @activation-key.json \  
https://satellite.example.com/katello/api/activation_keys/2 \  
| python3 -m json.tool
```

Example response:

```
{  
  "id": 2,  
  "name": "TestKey",  
  "description": "Just for testing",  
  "unlimited_hosts": false,  
  "auto_attach": true,  
  "content_view_id": null,  
  "environment_id": null,  
  "usage_count": 0,  
  "user_id": 3,  
  "max_hosts": 10,  
  "release_version": null,  
  "service_level": null,  
  "content_overrides": [  
  
  ],  
  "organization": {  
    "name": "Default Organization",  
    "label": "Default_Organization",  
    "id": 1  
  },  
  "created_at": "2017-02-16 12:37:47 UTC",  
  "updated_at": "2017-02-16 12:46:17 UTC",  
  "content_view": null,  
  "environment": null,  
  "products": null,  
  "host_collections": [  
  
  ],  
  "permissions": {  
    "view_activation_keys": true,  
    "edit_activation_keys": true,  
    "destroy_activation_keys": true  
  }  
}
```

3. In the Satellite web UI, verify the changes by navigating to **Content > Activation keys**.

2.1.4. Using the DELETE HTTP verb

To delete a resource, use the DELETE verb with an API route that includes the ID of the resource you want to delete.

Example

This example deletes the **TestKey** activation key which ID is 2:

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request DELETE \
--user sat_username:sat_password --insecure \
https://satellite.example.com/katello/api/activation_keys/2 \
| python3 -m json.tool
```

Example response:

```
output omitted
"started_at": "2017-02-16 12:58:17 UTC",
"ended_at": "2017-02-16 12:58:18 UTC",
"state": "stopped",
"result": "success",
"progress": 1.0,
"input": {
  "activation_key": {
    "id": 2,
    "name": "TestKey"
  }
}
output truncated
```

2.1.5. Relating API error messages to the API reference

The API uses a RAILS format to indicate an error:

```
Nested_Resource.Attribute_Name
```

This translates to the following format used in the API reference:

```
Resource[Nested_Resource_attributes][Attribute_Name_id]
```

2.2. UNDERSTANDING THE JSON RESPONSE FORMAT

Calls to the API return results in JSON format. The API call returns the result for a single-option response or for responses collections.



NOTE

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

JSON response format for single objects

You can use single-object JSON responses to work with a single object. API requests to a single object require the object's unique identifier **:id**.

This is an example of the format for a single-object request for the Satellite domain which ID is 23:

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password\  
https://satellite.example.com/api/domains/23 | python3 -m json.tool
```

Example response:

```
{  
  "id": 23,  
  "name": "qa.lab.example.com",  
  "fullname": "QA",  
  "dns_id": 10,  
  "created_at": "2013-08-13T09:02:31Z",  
  "updated_at": "2013-08-13T09:02:31Z"  
}
```

JSON response format for collections

Collections are a list of objects such as hosts and domains. The format for a collection JSON response consists of a metadata fields section and a results section.

This is an example of the format for a collection request for a list of Satellite domains:

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password\  
https://satellite.example.com/api/domains | python3 -m json.tool
```

Example response:

```
{  
  "total": 3,  
  "subtotal": 3,  
  "page": 1,  
  "per_page": 20,  
  "search": null,  
  "sort": {  
    "by": null,  
    "order": null  
  },  
  "results": [  
    {  
      "id": 23,  
      "name": "qa.lab.example.com",  
      "fullname": "QA",  
      "dns_id": 10,  
      "created_at": "2013-08-13T09:02:31Z",  
      "updated_at": "2013-08-13T09:02:31Z"  
    },  
    {  
      "id": 25,  
      "name": "sat.lab.example.com",  
      "fullname": "SATLAB",  
      "dns_id": 8,  
      "created_at": "2013-08-13T08:32:48Z",  
      "updated_at": "2013-08-13T08:32:48Z"  
    }  
  ]  
}
```

```

        "updated_at": "2013-08-14T07:04:03Z"
      },
      {
        "id": 32,
        "name": "hr.lab.example.com",
        "fullname": "HR",
        "dns_id": 8,
        "created_at": "2013-08-16T08:32:48Z",
        "updated_at": "2013-08-16T07:04:03Z"
      }
    ]
  }
}

```

The response metadata fields

API response uses the following metadata fields:

- **total** – The total number of objects without any search parameters.
- **subtotal** – The number of objects returned with the given search parameters. If there is no search, then subtotal is equal to total.
- **page** – The page number.
- **per_page** – The maximum number of objects returned per page.
- **limit** – The specified number of objects to return in a collection response.
- **offset** – The number of objects skipped before returning a collection.
- **search** – The search string based on **scoped_scoped** syntax.
- **sort**
 - **by** – Specifies by what field the API sorts the collection.
 - **order** – The sort order, either ASC for ascending or DESC for descending.
- **results** – The collection of objects.

CHAPTER 3. AUTHENTICATING API CALLS

Interaction with the Satellite API requires SSL authentication with Satellite Server CA certificate and authentication with valid Satellite user credentials. This chapter outlines the authenticating methods you can use.

3.1. SSL AUTHENTICATION OVERVIEW

Red Hat Satellite uses HTTPS, which provides a degree of encryption and identity verification when communicating with a Red Hat Satellite Server. Satellite 6.14 does not support non-SSL communications.

Each Red Hat Satellite Server uses a self-signed certificate. This certificate acts as both the server certificate to verify the encryption key and the certificate authority (CA) to trust the identity of Satellite Server.

3.1.1. Configuring SSL authentication

Use the following procedure to configure an SSL authentication for the API requests to Satellite Server.

Procedure

1. Obtain a certificate from the Satellite Server with which you want to communicate using one of the following options:

- If you execute the command from a remote server, obtain a certificate using SSH:

```
$ scp root@satellite.example.com:/var/www/html/pub/katello-server-ca.crt /etc/pki/ca-trust/source/anchors/satellite.example.com-katello-server-ca.crt
```

- If you execute the command directly on the Satellite Server, copy the certificate to the **/etc/pki/ca-trust/source/anchors** directory:

```
$ cp /var/www/html/pub/katello-server-ca.crt /etc/pki/ca-trust/source/anchors/satellite.example.com-katello-server-ca.crt
```

2. Add the certificate to the list of trusted CAs:

```
update-ca-trust extract
```

Verification

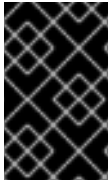
- Verify that the certificate is present in the NSS database by entering the API request without the **--cacert** option:

```
$ curl --request GET \
--user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts
```

3.2. HTTP AUTHENTICATION OVERVIEW

All requests to the Satellite API require a valid Satellite user name and password. The API uses HTTP

Basic Authentication to encode these credentials and add to the **Authorization** header. For more information about Basic Authentication, see [RFC 2617 HTTP Authentication: Basic and Digest Access Authentication](#). If a request does not include an appropriate **Authorization** header, the API returns a **401 Authorization Required** error



IMPORTANT

Basic authentication involves potentially sensitive information, for example, it sends passwords as plain text. The REST API requires HTTPS for transport level encryption of plain text requests.

Some base64 libraries break encoded credentials into multiple lines and terminate each line with a newline character. This invalidates the header and causes a faulty request. The Authorization header requires that the encoded credentials be on a single line within the header.

3.3. PERSONAL ACCESS TOKEN AUTHENTICATION OVERVIEW

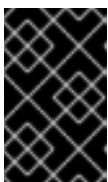
Red Hat Satellite supports Personal Access Tokens that you can use to authenticate API requests instead of using your password. You can set an expiration date for your Personal Access Token and you can revoke it if you decide it should expire before the expiration date.

3.3.1. Creating a Personal Access Token

Use this procedure to create a Personal Access Token.

Procedure

1. In the Satellite web UI, navigate to **Administer > Users**.
2. Select a user for which you want to create a Personal Access Token.
3. On the **Personal Access Tokens** tab, click **Add Personal Access Token**
4. Enter a **Name** for your Personal Access Token.
5. Optional: Select the **Expires** date to set an expiration date. If you do not set an expiration date, your Personal Access Token will never expire unless revoked.
6. Click Submit. You now have the Personal Access Token available to you on the **Personal Access Tokens** tab.



IMPORTANT

Ensure to store your Personal Access Token as you will not be able to access it again after you leave the page or create a new Personal Access Token. You can click **Copy to clipboard** to copy your Personal Access Token.

Verification

1. Make an API request to your Satellite Server and authenticate with your Personal Access Token:

```
# curl https://satellite.example.com/api/status --user
My_Username:My_Personal_Access_Token
```

2. You should receive a response with status **200**, for example:

```
{ "satellite_version": "6.14.0", "result": "ok", "status": 200, "version": "3.5.1.10", "api_version": 2 }
```

If you go back to **Personal Access Tokens** tab, you can see the updated **Last Used** time next to your Personal Access Token.

3.3.2. Revoking a Personal Access Token

Use this procedure to revoke a Personal Access Token before its expiration date.

Procedure

1. In the Satellite web UI, navigate to **Administer > Users**.
2. Select a user for which you want to revoke the Personal Access Token.
3. On the **Personal Access Tokens** tab, locate the Personal Access Token you want to revoke.
4. Click **Revoke** in the **Actions** column next to the Personal Access Token you want to revoke.

Verification

1. Make an API request to your Satellite Server and try to authenticate with the revoked Personal Access Token:

```
# curl https://satellite.example.com/api/status --user  
My_Username:My_Personal_Access_Token
```

2. You receive the following error message:

```
{  
  "error": { "message": "Unable to authenticate user My_Username" }  
}
```

3.4. OAUTH AUTHENTICATION OVERVIEW

As an alternative to basic authentication, you can use limited OAuth 1.0 authentication. This is sometimes referred to as 1-legged OAuth in version 1.0a of the protocol.

To view OAuth settings, in the Satellite web UI, navigate to **Administer > Settings > Authentication**. The **OAuth consumer key** is the token to be used by all OAuth clients.

Satellite stores OAuth settings in the `/etc/foreman/settings.yaml` file. Use the **satellite-installer** script to configure these settings, because Satellite overwrites any manual changes to this file when upgrading.

3.4.1. Configuring OAuth

To change the OAuth settings, enter the **satellite-installer** with the required options. Enter the following command to list all the OAuth related installer options:

```
# satellite-installer --full-help | grep oauth
```

Enabling OAuth mapping

By default, Satellite authorizes all OAuth API requests as the built-in anonymous API administrator account. Therefore, API responses include all Satellite data. However, you can also specify the Foreman user that makes the request and restrict access to data to that user.

To enable OAuth user mapping, enter the following command:

```
# satellite-installer --foreman-oauth-map-users true
```



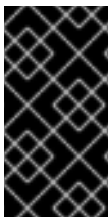
IMPORTANT

Satellite does not sign the header in an OAuth request. Anyone with a valid consumer key can impersonate any Foreman user.

3.4.2. OAuth request format

Every OAuth API request requires the **FOREMAN-USER** header with the login of an existing Foreman user and the **Authorization** header in the following format:

```
--header 'FOREMAN-USER: sat_username' \
--header 'Authorization: OAuth
oauth_version="1.0",oauth_consumer_key="secretkey",oauth_signature_method="hmac-
sha1",oauth_timestamp=timestamp,oauth_signature=signature'
```



IMPORTANT

Use an OAuth client library to construct all OAuth parameters. For an example that uses the **requests_oauthlib** Python module, see [How to execute an API call using the OAuth authentication method via python script in Red Hat Satellite 6?](#) in the Red Hat Knowledgebase.

Example

This example lists architectures using OAuth for authentication. The request uses a *sat_username* username in the **FOREMAN-USER** header. With the **--foreman-oauth-map-users** set to **true**, the response includes only architectures that the user has access to view. The signature reflects every parameter, HTTP method, and URI change.

Example request:

```
$ curl 'https://satellite.example.com/api/architectures' \
--header 'Content-Type: application/json' \
--header 'Accept:application/json' \
--header 'FOREMAN-USER: sat_username' \
--header 'Authorization: OAuth
oauth_version="1.0",oauth_consumer_key="secretkey",oauth_signature_method="hmac-
sha1",oauth_timestamp=1321473112,oauth_signature=ll8hR8/ogj/XVuOqMPB9qNjSy6E='
```

Additional resources

- [Documentation for OAuth Core 1.0 Revision A](#)

CHAPTER 4. API REQUESTS IN DIFFERENT LANGUAGES

This chapter outlines sending API requests to Red Hat Satellite with `curl`, Ruby, and Python and provides examples.

4.1. API REQUESTS WITH CURL

This section outlines how to use **curl** with the Satellite API to perform various tasks.

Red Hat Satellite requires the use of HTTPS, and by default a certificate for host identification. If you have not added the Satellite Server certificate as described in [Section 3.1, "SSL authentication overview"](#), then you can use the **--insecure** option to bypass certificate checks.

For user authentication, you can use the **--user** option to provide Satellite user credentials in the form **--user *username:password*** or, if you do not include the password, the command prompts you to enter it. To reduce security risks, do not include the password as part of the command, because it then becomes part of your shell history. Examples in this section include the password only for the sake of simplicity.

Be aware that if you use the **--silent** option, **curl** does not display a progress meter or any error messages.

Examples in this chapter use the Python **json.tool** module to format the output.

4.1.1. Passing JSON data to the API request

You can pass data to Satellite Server with the API request. The data must be in JSON format. When specifying JSON data with the **--data** option, you must set the following HTTP headers with the **--header** option:

```
--header "Accept:application/json" \
--header "Content-Type:application/json"
```

Use one of the following options to include data with the **--data** option:

1. The quoted JSON formatted data enclosed in curly braces **{}**. When passing a value for a JSON type parameter, you must escape quotation marks **"** with backslashes ****. For example, within curly braces, you must format **"Example JSON Variable"** as **"\Example JSON Variable\"**:

```
--data {"id":44, "smart_class_parameter":{"override":"true", "parameter_type":"json",
"default_value":{"GRUB_CMDLINE_LINUX": {"audit":"1","crashkernel":"true"}}}}
```

2. The unquoted JSON formatted data enclosed in a file and specified by the **@** sign and the filename. For example:

```
--data @file.json
```

Using external files for JSON formatted data has the following advantages:

- You can use your favorite text editor.
- You can use syntax checker to find and avoid mistakes.
- You can use tools to check the validity of JSON data or to reformat it.

Validating a JSON file

Use the **json_verify** tool to check the validity of a JSON file:

```
$ json_verify < test_file.json
```

4.1.2. Retrieving a list of resources

This section outlines how to use **curl** with the Satellite 6 API to request information from your Satellite deployment. These examples include both requests and responses. Expect different results for each deployment.



NOTE

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Listing users

This example is a basic request that returns a list of Satellite resources, Satellite users in this case. Such requests return a list of data wrapped in metadata, while other request types only return the actual object.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password\  
https://satellite.example.com/api/users | python3 -m json.tool
```

Example response:

```
{  
  "page": 1,  
  "per_page": 20,  
  "results": [  
    {  
      "admin": false,  
      "auth_source_id": 1,  
      "auth_source_name": "Internal",  
      "created_at": "2018-09-21 08:59:22 UTC",  
      "default_location": null,  
      "default_organization": null,  
      "description": "",  
      "effective_admin": false,  
      "firstname": "",  
      "id": 5,  
      "last_login_on": "2018-09-21 09:03:25 UTC",  
      "lastname": "",  
      "locale": null,  
      "locations": [],  
      "login": "test",  
      "mail": "example@domain.com",  
      "organizations": [  
        {
```

```

        "id": 1,
        "name": "Default Organization"
      }
    ],
    "ssh_keys": [],
    "timezone": null,
    "updated_at": "2018-09-21 09:04:45 UTC"
  },
  {
    "admin": true,
    "auth_source_id": 1,
    "auth_source_name": "Internal",
    "created_at": "2018-09-20 07:09:41 UTC",
    "default_location": null,
    "default_organization": {
      "description": null,
      "id": 1,
      "name": "Default Organization",
      "title": "Default Organization"
    },
    "description": "",
    "effective_admin": true,
    "firstname": "Admin",
    "id": 4,
    "last_login_on": "2018-12-07 07:31:09 UTC",
    "lastname": "User",
    "locale": null,
    "locations": [
      {
        "id": 2,
        "name": "Default Location"
      }
    ],
    "login": "admin",
    "mail": "root@example.com",
    "organizations": [
      {
        "id": 1,
        "name": "Default Organization"
      }
    ],
    "ssh_keys": [],
    "timezone": null,
    "updated_at": "2018-11-14 08:19:46 UTC"
  }
],
"search": null,
"sort": {
  "by": null,
  "order": null
},
"subtotal": 2,
"total": 2
}

```

4.1.3. Creating and modifying resources

This section outlines how to use **curl** with the Satellite 6 API to manipulate resources on the Satellite Server. These API calls require that you pass data in **json** format with the API call. For more information, see [Section 4.1.1, “Passing JSON data to the API request”](#).

Creating a user

This example creates a user using **--data** option to provide required information.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request POST \
--user sat_username:sat_password --insecure \
--data '{"firstname":"'Test
Name',"mail":"'test@example.com',"login":"'test_user',"password":"'password123',"auth_sour
ce_id":1}' \
https://satellite.example.com/api/users | python3 -m json.tool
```

Modifying a user

This example modifies first name and login of the **test_user** that was created in [Creating a user](#).

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request PUT \
--user sat_username:sat_password --insecure \
--data '{"firstname":"'New Test
Name',"mail":"'test@example.com',"login":"'new_test_user',"password":"'password123',"auth
_source_id":1}' \
https://satellite.example.com/api/users/8 | python3 -m json.tool
```

4.2. API REQUESTS WITH RUBY

This section outlines how to use Ruby with the Satellite API to perform various tasks.



IMPORTANT

These are example scripts and commands. Ensure you review these scripts carefully before use, and replace any variables, user names, passwords, and other information to suit your own deployment.

4.2.1. Creating objects using Ruby

This script connects to the Red Hat Satellite 6 API and creates an organization, and then creates three environments in the organization. If the organization already exists, the script uses that organization. If any of the environments already exist in the organization, the script raises an error and quits.

```
#!/usr/bin/ruby

require 'rest-client'
require 'json'
```

```

url = 'https://satellite.example.com/api/v2/'
katello_url = "#{url}/katello/api/v2/"

$username = 'admin'
$password = 'changeme'

org_name = "MyOrg"
environments = [ "Development", "Testing", "Production" ]

# Performs a GET using the passed URL location
def get_json(location)
  response = RestClient::Request.new(
    :method => :get,
    :url => location,
    :user => $username,
    :password => $password,
    :headers => { :accept => :json,
                  :content_type => :json }
  ).execute
  JSON.parse(response.to_str)
end

# Performs a POST and passes the data to the URL location
def post_json(location, json_data)
  response = RestClient::Request.new(
    :method => :post,
    :url => location,
    :user => $username,
    :password => $password,
    :headers => { :accept => :json,
                  :content_type => :json },
    :payload => json_data
  ).execute
  JSON.parse(response.to_str)
end

# Creates a hash with ids mapping to names for an array of records
def id_name_map(records)
  records.inject({}) do |map, record|
    map.update(record["id"] => record["name"])
  end
end

# Get list of existing organizations
orgs = get_json("#{katello_url}/organizations")
org_list = id_name_map(orgs['results'])

if !org_list.has_value?(org_name)
  # If our organization is not found, create it
  puts "Creating organization: \t#{org_name}"
  org_id = post_json("#{katello_url}/organizations", JSON.generate({"name"=> org_name}))["id"]
else
  # Our organization exists, so let's grab it
  org_id = org_list.key(org_name)
  puts "Organization \"#{org_name}\" exists"
end

```

```

end

# Get list of organization's lifecycle environments
envs = get_json("#{katello_url}/organizations/#{org_id}/environments")
env_list = id_name_map(envs['results'])
prior_env_id = env_list.key("Library")

# Exit the script if at least one life cycle environment already exists
environments.each do |e|
  if env_list.has_value?(e)
    puts "ERROR: One of the Environments is not unique to organization"
    exit
  end
end

# Create life cycle environments
environments.each do |environment|
  puts "Creating environment: \t#{environment}"
  prior_env_id = post_json("#{katello_url}/organizations/#{org_id}/environments",
    JSON.generate({"name" => environment, "organization_id" => org_id, "prior_id" => prior_env_id}))
  ["id"]
end

```

4.2.2. Using apipie bindings with Ruby

Apipie bindings are the Ruby bindings for apipie documented API calls. They fetch and cache the API definition from Satellite and then generate API calls on demand. This example creates an organization, and then creates three environments in the organization. If the organization already exists, the script uses that organization. If any of the environments already exist in the organization, the script raises an error and quits.

```

#!/usr/bin/tfm-ruby

require 'apipie-bindings'

org_name = "MyOrg"
environments = [ "Development", "Testing", "Production" ]

# Create an instance of apipie bindings
@api = ApipieBindings::API.new({
  :uri => 'https://satellite.example.com/',
  :username => 'admin',
  :password => 'changeme',
  :api_version => 2
})

# Performs an API call with default options
def call_api(resource_name, action_name, params = {})
  http_headers = {}
  apipie_options = { :skip_validation => true }
  @api.resource(resource_name).call(action_name, params, http_headers, apipie_options)
end

# Creates a hash with IDs mapping to names for an array of records
def id_name_map(records)

```

```

records.inject({}) do |map, record|
  map.update(record['id'] => record['name'])
end
end

# Get list of existing organizations
orgs = call_api(:organizations, :index)
org_list = id_name_map(orgs['results'])

if !org_list.has_value?(org_name)
  # If our organization is not found, create it
  puts "Creating organization: \#{org_name}"
  org_id = call_api(:organizations, :create, {'organization' => { :name => org_name }})['id']
else
  # Our organization exists, so let's grab it
  org_id = org_list.key(org_name)
  puts "Organization \#{org_name}\" exists"
end

# Get list of organization's life cycle environments
envs = call_api(:lifecycle_environments, :index, {'organization_id' => org_id})
env_list = id_name_map(envs['results'])
prior_env_id = env_list.key("Library")

# Exit the script if at least one life cycle environment already exists
environments.each do |e|
  if env_list.has_value?(e)
    puts "ERROR: One of the Environments is not unique to organization"
    exit
  end
end

# Create life cycle environments
environments.each do |environment|
  puts "Creating environment: \#{environment}"
  prior_env_id = call_api(:lifecycle_environments, :create, {"name" => environment, "organization_id"
=> org_id, "prior_id" => prior_env_id })['id']
end

```

4.3. API REQUESTS WITH PYTHON

This section outlines how to use Python with the Satellite API to perform various tasks.



IMPORTANT

These are example scripts and commands. Ensure you review these scripts carefully before use, and replace any variables, user names, passwords, and other information to suit your own deployment.

Example scripts in this section do not use SSL verification for interacting with the REST API.

4.3.1. Creating objects using Python

This script connects to the Red Hat Satellite 6 API and creates an organization, and then creates three environments in the organization. If the organization already exists, the script uses that organization. If any of the environments already exist in the organization, the script raises an error and quits.

Python 2 example

```
#!/usr/bin/python

import json
import sys

try:
    import requests
except ImportError:
    print "Please install the python-requests module."
    sys.exit(-1)

# URL to your Satellite 6 server
URL = "https://satellite.example.com"
# URL for the API to your deployed Satellite 6 server
SAT_API = "%s/katello/api/v2/" % URL
# Katello-specific API
KATELLO_API = "%s/katello/api/" % URL
POST_HEADERS = {'content-type': 'application/json'}
# Default credentials to login to Satellite 6
USERNAME = "admin"
PASSWORD = "changeme"
# Ignore SSL for now
SSL_VERIFY = False

# Name of the organization to be either created or used
ORG_NAME = "MyOrg"
# Name for life cycle environments to be either created or used
ENVIRONMENTS = ["Development", "Testing", "Production"]

def get_json(location):
    """
    Performs a GET using the passed URL location
    """

    r = requests.get(location, auth=(USERNAME, PASSWORD), verify=SSL_VERIFY)

    return r.json()

def post_json(location, json_data):
    """
    Performs a POST and passes the data to the URL location
    """

    result = requests.post(
        location,
        data=json_data,
        auth=(USERNAME, PASSWORD),
        verify=SSL_VERIFY,
```

```

        headers=POST_HEADERS)

    return result.json()

def main():
    """
    Main routine that creates or re-uses an organization and
    life cycle environments. If life cycle environments already
    exist, exit out.
    """

    # Check if our organization already exists
    org = get_json(SAT_API + "organizations/" + ORG_NAME)

    # If our organization is not found, create it
    if org.get('error', None):
        org_id = post_json(
            SAT_API + "organizations/",
            json.dumps({"name": ORG_NAME}))["id"]
        print "Creating organization: \t" + ORG_NAME
    else:
        # Our organization exists, so let's grab it
        org_id = org['id']
        print "Organization '%s' exists." % ORG_NAME

    # Now, let's fetch all available life cycle environments for this org...
    envs = get_json(
        SAT_API + "organizations/" + str(org_id) + "/environments/")

    # ... and add them to a dictionary, with respective 'Prior' environment
    prior_env_id = 0
    env_list = {}
    for env in envs['results']:
        env_list[env['id']] = env['name']
        prior_env_id = env['id'] if env['name'] == "Library" else prior_env_id

    # Exit the script if at least one life cycle environment already exists
    if all(environment in env_list.values() for environment in ENVIRONMENTS):
        print "ERROR: One of the Environments is not unique to organization"
        sys.exit(-1)

    # Create life cycle environments
    for environment in ENVIRONMENTS:
        new_env_id = post_json(
            SAT_API + "organizations/" + str(org_id) + "/environments/",
            json.dumps(
                {
                    "name": environment,
                    "organization_id": org_id,
                    "prior": prior_env_id
                }
            ))["id"]

        print "Creating environment: \t" + environment
        prior_env_id = new_env_id

```

```
if __name__ == "__main__":
    main()
```

4.3.2. Requesting information from the API using Python

This is an example script that uses Python for various API requests.

Python 2 example

```
#!/usr/bin/python
import json
import sys
try:
    import requests
except ImportError:
    print "Please install the python-requests module."
    sys.exit(-1)

SAT_API = 'https://satellite.example.com/api/v2/'
USERNAME = "admin"
PASSWORD = "password"
SSL_VERIFY = False # Ignore SSL for now

def get_json(url):
    # Performs a GET using the passed URL location
    r = requests.get(url, auth=(USERNAME, PASSWORD), verify=SSL_VERIFY)
    return r.json()

def get_results(url):
    jsn = get_json(url)
    if jsn.get('error'):
        print "Error: " + jsn['error']['message']
    else:
        if jsn.get('results'):
            return jsn['results']
        elif 'results' not in jsn:
            return jsn
        else:
            print "No results found"
    return None

def display_all_results(url):
    results = get_results(url)
    if results:
        print json.dumps(results, indent=4, sort_keys=True)

def display_info_for_hosts(url):
    hosts = get_results(url)
    if hosts:
        for host in hosts:
            print "ID: %-10d Name: %-30s IP: %-20s OS: %-30s" % (host['id'], host['name'], host['ip'],
            host['operatingsystem_name'])

def main():
```

```

host = 'satellite.example.com'
print "Displaying all info for host %s ..." % host
display_all_results(SAT_API + 'hosts/' + host)

print "Displaying all facts for host %s ..." % host
display_all_results(SAT_API + 'hosts/%s/facts' % host)

host_pattern = 'example'
print "Displaying basic info for hosts matching pattern '%s'..." % host_pattern
display_info_for_hosts(SAT_API + 'hosts?search=' + host_pattern)

environment = 'production'
print "Displaying basic info for hosts in environment %s..." % environment
display_info_for_hosts(SAT_API + 'hosts?search=environment=' + environment)

model = 'RHEV Hypervisor'
print "Displaying basic info for hosts with model name %s..." % model
display_info_for_hosts(SAT_API + 'hosts?search=model=' + model + '')

if __name__ == "__main__":
    main()

```

Python 3 example

```

#!/usr/bin/env python3

import json
import sys

try:
    import requests
except ImportError:
    print("Please install the python-requests module.")
    sys.exit(-1)

SAT = "satellite.example.com"
# URL for the API to your deployed Satellite 6 server
SAT_API = f"https://{SAT}/api/"
KATELLO_API = f"https://{SAT}/katello/api/v2/"

POST_HEADERS = {'content-type': 'application/json'}
# Default credentials to login to Satellite 6
USERNAME = "admin"
PASSWORD = "password"
# Ignore SSL for now
SSL_VERIFY = False
#SSL_VERIFY = "/path/to/CA-certificate.crt" # Put the path to your CA certificate here to allow
SSL_VERIFY

def get_json(url):
    # Performs a GET using the passed URL location
    r = requests.get(url, auth=(USERNAME, PASSWORD), verify=SSL_VERIFY)
    return r.json()

def get_results(url):

```

```

jsn = get_json(url)
if jsn.get('error'):
    print("Error: " + jsn['error']['message'])
else:
    if jsn.get('results'):
        return jsn['results']
    elif 'results' not in jsn:
        return jsn
    else:
        print("No results found")
return None

def display_all_results(url):
    results = get_results(url)
    if results:
        print(json.dumps(results, indent=4, sort_keys=True))

def display_info_for_hosts(url):
    hosts = get_results(url)
    if hosts:
        print(f'{{"ID":10}}{{"Name":40}}{{"IP":30}}{{"Operating System":30}}')
        for host in hosts:
            print(f'{{str(host["id"]):10}}{{host["name"]:40}}{{str(host["ip"]):30}}
{{str(host["operatingsystem_name"]):30}}')

def display_info_for_subs(url):
    subs = get_results(url)
    if subs:
        print(f'{{"ID":10}}{{"Name":90}}{{"Start Date":30}}')
        for sub in subs:
            print(f'{{str(sub["id"]):10}}{{sub["name"]:90}}{{str(sub["start_date"]):30}}')

def main():
    host = SAT
    print(f"Displaying all info for host {host} ...")
    display_all_results(SAT_API + 'hosts/' + host)

    print(f"Displaying all facts for host {host} ...")
    display_all_results(SAT_API + f'hosts/{host}/facts')

    host_pattern = 'example'
    print(f"Displaying basic info for hosts matching pattern '{host_pattern}'...")
    display_info_for_hosts(SAT_API + 'hosts?per_page=1&search=name~' + host_pattern)

    print(f"Displaying basic info for subscriptions")
    display_info_for_subs(KATELLO_API + 'subscriptions')

    environment = 'production'
    print(f"Displaying basic info for hosts in environment {environment}...")
    display_info_for_hosts(SAT_API + 'hosts?search=environment=' + environment)

if __name__ == "__main__":
    main()

```

CHAPTER 5. USING THE RED HAT SATELLITE API

This chapter provides a range of examples of how to use the Red Hat Satellite API to perform different tasks. You can use the API on Satellite Server via HTTPS on port 443, or on Capsule Server via HTTPS on port 8443.

You can address these different port requirements within the script itself. For example, in Ruby, you can specify the Satellite and Capsule URLs as follows:

```
url = 'https://satellite.example.com/api/v2/'
capsule_url = 'https://capsule.example.com:8443/api/v2/'
katello_url = 'https://satellite.example.com/katello/api/v2/'
```

For the host that is subscribed to Satellite Server or Capsule Server, you can determine the correct port required to access the API from the `/etc/rhsm/rhsm.conf` file, in the port entry of the **[server]** section. You can use these values to fully automate your scripts, removing any need to verify which ports to use.

This chapter uses **curl** for sending API requests. For more information, see [Section 4.1, “API requests with curl”](#).

Examples in this chapter use the Python **json.tool** module to format the output.

5.1. WORKING WITH HOSTS



NOTE

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Listing hosts

This example returns a list of Satellite hosts.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts | python3 -m json.tool
```

Example response:

```
{
  ...
  "total" => 2,
  "subtotal" => 2,
  "page" => 1,
  "per_page" => 1000,
  "search" => nil,
  "sort" => {
    "by" => nil,
    "order" => nil
  },
}
```

```
"results" => [
  ...
]
```

Requesting information for a host

This request returns information for the host **satellite.example.com**.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts/satellite.example.com \
| python3 -m json.tool
```

Example response:

```
{
  "all_puppetclasses": [],
  "architecture_id": 1,
  "architecture_name": "x86_64",
  "build": false,
  "capabilities": [
    "build"
  ],
  "certname": "satellite.example.com",
  "comment": null,
  "compute_profile_id": null,
  ...
}
```

Listing host facts

This request returns all facts for the host **satellite.example.com**.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts/satellite.example.com/facts \
| python3 -m json.tool
```

Example response:

```
{
  ...
  "results": {
    "satellite.example.com": {
      "augeasversion": "1.0.0",
      "bios_release_date": "01/01/2007",
      "bios_version": "0.5.1",
      "blockdevice_sr0_size": "1073741312",
      "facterversion": "1.7.6",
      ...
    }
  }
}
```

Searching for hosts with matching patterns

This query returns all hosts that match the pattern "example".

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password\  
https://satellite.example.com/api/v2/hosts?search=example \  
| python3 -m json.tool
```

Example response:

```
{  
  ...  
  "results": [  
    {  
      "name": "satellite.example.com",  
      ...  
    }  
  ],  
  "search": "example",  
  ...  
}
```

Searching for hosts in an environment

This query returns all hosts in the **production** environment.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password\  
https://satellite.example.com/api/v2/hosts?search=environment=production \  
| python3 -m json.tool
```

Example response:

```
{  
  ...  
  "results": [  
    {  
      "environment_name": "production",  
      "name": "satellite.example.com",  
      ...  
    }  
  ],  
  "search": "environment=production",  
  ...  
}
```

Searching for hosts with a specific fact value

This query returns all hosts with a model name **RHEV Hypervisor**.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts?search=model=\"RHEV+Hypervisor\" \
| python3 -m json.tool
```

Example response:

```
{
  ...
  "results": [
    {
      "model_id": 1,
      "model_name": "RHEV Hypervisor",
      "name": "satellite.example.com",
      ...
    }
  ],
  "search": "model=\"RHEV Hypervisor\"",
  ...
}
```

Deleting a host

This request deletes a host with a name *host1.example.com*.

Example request:

```
$ curl --request DELETE --insecure --user sat_username:sat_password \
https://satellite.example.com/api/v2/hosts/host1.example.com \
| python3 -m json.tool
```

Downloading a full boot disk image

This request downloads a full boot disk image for a host by its ID.

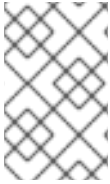
Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/bootdisk/api/hosts/host_ID?full=true \
--output image.iso
```

5.2. WORKING WITH LIFE CYCLE ENVIRONMENTS

Satellite divides application life cycles into life cycle environments, which represent each stage of the application life cycle. Life cycle environments are linked to from an environment path. To create linked life cycle environments with the API, use the **prior_id** parameter.

You can find the built-in API reference for life cycle environments at **https://satellite.example.com/apidoc/v2/lifecycle_environments.html**. The API routes include **/katello/api/environments** and **/katello/api/organizations/:organization_id/environments**.

**NOTE**

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Listing life cycle environments

Use this API call to list all the current life cycle environments on your Satellite for the default organization with ID **1**.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request GET --user sat_username:sat_password --insecure \
https://satellite.example.com/katello/api/organizations/1/environments \
| python3 -m json.tool`
```

Example response:

```
output omitted
"description": null,
"id": 1,
"label": "Library",
"library": true,
"name": "Library",
"organization": {
  "id": 1,
  "label": "Default_Organization",
  "name": "Default Organization"
},
"permissions": {
  "destroy_lifecycle_environments": false,
  "edit_lifecycle_environments": true,
  "promote_or_remove_content_views_to_environments": true,
  "view_lifecycle_environments": true
},
"prior": null,
"successor": null,
output truncated
```

Creating linked life cycle environments

Use this example to create a path of life cycle environments.

This procedure uses the default Library environment with ID **1** as the starting point for creating life cycle environments.

1. Choose an existing life cycle environment that you want to use as a starting point. List the environment using its ID, in this case, the environment with ID **1**:

Example request:

```
$ curl --request GET --user sat_username:sat_password --insecure \
https://satellite.example.com/katello/api/environments/1 \
| python3 -m json.tool
```

Example response:

```
output omitted
"id": 1,
"label": "Library",
output omitted
"prior": null,
"successor": null,
output truncated
```

2. Create a JSON file, for example, **life-cycle.json**, with the following content:

```
{"organization_id":1,"label":"api-dev","name":"API Development","prior":1}
```

3. Create a life cycle environment using the **prior** option set to **1**.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request POST --user sat_username:sat_password --insecure \
--data @life-cycle.json \
https://satellite.example.com/katello/api/environments \
| python3 -m json.tool
```

Example response:

```
output omitted
"description": null,
"id": 2,
"label": "api-dev",
"library": false,
"name": "API Development",
"organization": {
  "id": 1,
  "label": "Default_Organization",
  "name": "Default Organization"
},
"permissions": {
  "destroy_lifecycle_environments": true,
  "edit_lifecycle_environments": true,
  "promote_or_remove_content_views_to_environments": true,
  "view_lifecycle_environments": true
},
"prior": {
  "id": 1,
  "name": "Library"
},
output truncated
```

In the command output, you can see the ID for this life cycle environment is **2**, and the life cycle environment prior to this one is **1**. Use the life cycle environment with ID **2** to create a successor to this environment.

4. Edit the previously created **life-cycle.json** file, updating the **label**, **name**, and **prior** values.

```
{ "organization_id":1,"label":"api-qa","name":"API QA","prior":2}
```

5. Create a life cycle environment, using the **prior** option set to **2**.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request POST --user sat_username:sat_password --insecure \
--data @life-cycle.json \
https://satellite.example.com/katello/api/environments \
| python3 -m json.tool
```

Example response:

```
output omitted
"description": null,
"id": 3,
"label": "api-qa",
"library": false,
"name": "API QA",
"organization": {
  "id": 1,
  "label": "Default_Organization",
  "name": "Default Organization"
},
"permissions": {
  "destroy_lifecycle_environments": true,
  "edit_lifecycle_environments": true,
  "promote_or_remove_content_views_to_environments": true,
  "view_lifecycle_environments": true
},
"prior": {
  "id": 2,
  "name": "API Development"
},
"successor": null,
output truncated
```

In the command output, you can see the ID for this life cycle environment is **3**, and the life cycle environment prior to this one is **2**.

Updating a life cycle environment

You can update a life cycle environment using a PUT command.

This example request updates a description of the life cycle environment with ID **3**.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request POST --user sat_username:sat_password --insecure \
--data '{"description":"Quality Acceptance Testing"}' \
https://satellite.example.com/katello/api/environments/3 \
| python3 -m json.tool
```

Example response:

```
output omitted
"description": "Quality Acceptance Testing",
"id": 3,
"label": "api-qa",
"library": false,
"name": "API QA",
"organization": {
  "id": 1,
  "label": "Default_Organization",
  "name": "Default Organization"
},
"permissions": {
  "destroy_lifecycle_environments": true,
  "edit_lifecycle_environments": true,
  "promote_or_remove_content_views_to_environments": true,
  "view_lifecycle_environments": true
},
"prior": {
  "id": 2,
  "name": "API Development"
},
output truncated
```

Deleting a life cycle environment

You can delete a life cycle environment provided it has no successor. Therefore, delete them in reverse order using a command in the following format:

Example request:

```
$ curl --request DELETE --user sat_username:sat_password --insecure \
https://satellite.example.com/katello/api/environments/:id
```

5.3. UPLOADING CONTENT TO THE SATELLITE SERVER

This section outlines how to use the Satellite 6 API to upload and import large files to your Satellite Server. This process involves four steps:

1. Create an upload request.
2. Upload the content.
3. Import the content.
4. Delete the upload request.

The maximum file size that you can upload is 2MB. For information about uploading larger content, see [Uploading content larger than 2 MB](#).

Procedure

1. Assign the package name to the variable **name**:

Example request:

```
$ export name=jq-1.6-2.el7.x86_64.rpm
```

2. Assign the checksum of the file to the variable **checksum**:

Example request:

```
$ export checksum=$(sha256sum $name|cut -c 1-65)
```

3. Assign the file size to the variable **size**:

Example request:

```
$ export size=$(du -bs $name|cut -f 1)
```

4. The following command creates a new upload request and returns the upload ID of the request using **size** and **checksum**.

Example request:

```
$ curl -H 'Content-Type: application/json' -X POST -k \
-u sat_username:sat_password -d '{"size": "$size", \
  "checksum": "$checksum"}' \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads
```

where 76, in this case, is an example Repository ID.

Example request:

```
{"upload_id": "37eb5900-597e-4ac3-9bc5-2250c302fdc4"}
```

5. Assign the upload ID to the variable **upload_id**:

Example request:

```
$ export upload_id=37eb5900-597e-4ac3-9bc5-2250c302fdc4
```

6. Assign the path of the package you want to upload to the variable **path**:

```
$ export path=/root/jq/jq-1.6-2.el7.x86_64.rpm
```

7. Upload your content. Ensure you use the correct MIME type when you upload data. The API uses the application/json MIME type for the majority of requests to Satellite 6. Combine the upload_id, MIME type, and other parameters to upload content.

Example request:

```
$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data -X PUT --data-urlencode size=$size --data-urlencode \
offset=0 \
```

```
--data-urlencode content@${path} \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads/$upload_id
```

- After you have uploaded the content to the Satellite Server, you need to import it into the appropriate repository. Until you complete this step, the Satellite Server does not detect the new content.

Example request:

```
$ curl -H "Content-Type:application/json" -X PUT -u \
sat_username:sat_password -k -d \
{"uploads":[{"id": "$upload_id", "name": "$name", \
"checksum": "$checksum"}]} \
https://$(hostname -f)/katello/api/v2/repositories/76/import_uploads
```

- After you have successfully uploaded and imported your content, you can delete the upload request. This frees any temporary disk space that data is using during the upload.

Example request:

```
$ curl -H 'Content-Type: application/json' -X DELETE -k \
-u sat_username:sat_password -d "{}" \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads/$upload_id
```

Uploading content larger than 2 MB

The following example demonstrates how to split a large file into chunks, create an upload request, upload the individual files, import them to Satellite, and then delete the upload request. Note that this example uses sample content, host names, user names, repository ID, and file names.

- Assign the package name to the variable **name**:

```
$ export name=bpftool-3.10.0-1160.2.1.el7.centos.plus.x86_64.rpm
```

- Assign the checksum of the file to the variable **checksum**:

```
$ export checksum=$(sha256sum $name|cut -c 1-65)
```

- Assign the file size to the variable **size**:

```
$ export size=$(du -bs $name|cut -f 1)
```

- The following command creates a new upload request and returns the upload ID of the request using **size** and **checksum**.

Example request:

```
$ curl -H 'Content-Type: application/json' -X POST -k \
-u sat_username:sat_password -d '{"size": "$size", \
"checksum": "$checksum"}' \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads
```

where 76, in this case, is an example Repository ID.

Example output

```
{"upload_id":"37eb5900-597e-4ac3-9bc5-2250c302fdc4"}
```

5. Assign the upload ID to the variable **upload_id**:

```
$ export upload_id=37eb5900-597e-4ac3-9bc5-2250c302fdc4
```

6. Split the file in 2MB chunks:

```
$ split --bytes 2MB --numeric-suffixes \
--suffix-length=1 bpftool-3.10.0-1160.2.1.el7.centos.plus.x86_64.rpm bpftool
```

Example output

```
$ ls bpftool[0-9] -l
-rw-r--r--. 1 root root 2000000 Mar 31 14:15 bpftool0
-rw-r--r--. 1 root root 2000000 Mar 31 14:15 bpftool1
-rw-r--r--. 1 root root 2000000 Mar 31 14:15 bpftool2
-rw-r--r--. 1 root root 2000000 Mar 31 14:15 bpftool3
-rw-r--r--. 1 root root 868648 Mar 31 14:15 bpftool4
```

7. Assign the prefix of the split files to the variable **path**.

```
$ export path=/root/tmp/bpftool
```

8. Upload the file chunks. The offset starts at 0 for the first chunk and increases by 2000000 for each file. Note the use of the offset parameter and how it relates to the file size. Note also that the indexes are used after the path variable, for example, `${path}0`, `${path}1`.

Example requests:

```
$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data \
-X PUT --data-urlencode size=$size --data-urlencode offset=0 \
--data-urlencode content@${path}0 https://$(hostname -
f)/katello/api/v2/repositories/76/content_uploads/$upload_id

$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data \
-X PUT --data-urlencode size=$size --data-urlencode offset=2000000 \
--data-urlencode content@${path}1 https://$(hostname -
f)/katello/api/v2/repositories/76/content_uploads/$upload_id

$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data \
-X PUT --data-urlencode size=$size --data-urlencode offset=4000000 \
--data-urlencode content@${path}2 https://$(hostname -
f)/katello/api/v2/repositories/76/content_uploads/$upload_id

$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data \
-X PUT --data-urlencode size=$size --data-urlencode offset=6000000 \
--data-urlencode content@${path}3 https://$(hostname -
f)/katello/api/v2/repositories/76/content_uploads/$upload_id
```

```
$ curl -u sat_username:sat_password -H Accept:application/json -H \
Content-Type:multipart/form-data \
-X PUT --data-urlencode size=$size --data-urlencode offset=8000000 \
--data-urlencode content@${path}4 https://$(hostname -
f)/katello/api/v2/repositories/76/content_uploads/$upload_id
```

9. Import the complete upload to the repository:

```
$ curl -H "Content-Type:application/json" -X PUT -u \
sat_username:sat_password -k -d \
{"uploads":[{"id": "$upload_id", \
"name": "$name", "checksum": "$checksum"}]} \
https://$(hostname -f)/katello/api/v2/repositories/76/import_uploads
```

10. Delete the upload request:

```
$ curl -H 'Content-Type: application/json' -X DELETE -k \
-u sat_username:sat_password -d "{}" \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads/$upload_id
```

Uploading duplicate content

Note that if you try to upload duplicate content using:

Example request:

```
$ curl -H 'Content-Type: application/json' -X POST -k \
-u sat_username:sat_password -d '{"size": "$size", "checksum": "$checksum"}' \
https://$(hostname -f)/katello/api/v2/repositories/76/content_uploads
```

The call will return a content unit ID instead of an upload ID, similar to this:

```
{"content_unit_href":"/pulp/api/v3/content/file/files/c1bcdfb8-d840-4604-845e-86e82454c747/"}
```

You can copy this output and call import uploads directly to add the content to a repository:

Example request:

```
$ curl -H "Content-Type:application/json" -X PUT -u \
sat_username:sat_password -k -d \
{"uploads":[{"content_unit_id": "/pulp/api/v3/content/file/files/c1bcdfb8-d840-4604-845e-
86e82454c747/", \
"name": "$name", "checksum": "$checksum"}]} https://$(hostname -
f)/katello/api/v2/repositories/76/import_uploads
```

Note that the call changes from using **upload_id** to using **content_unit_id**.

5.4. APPLYING ERRATA TO A HOST OR HOST COLLECTION

You can use the API to apply errata to a host, host group, or host collection. The following is the basic syntax of a PUT request:

```
$ curl --header "Accept:application/json" \
```

```
--header "Content-Type:application/json" --request PUT \
--user sat_username:sat_password --insecure \
--data json-formatted-data https://satellite7.example.com
```

You can browse the built in API doc to find a URL to use for applying Errata. You can use the Satellite web UI to help discover the format for the search query. Navigate to **Hosts > Host Collections** and select a host collection. Go to **Collection Actions > Errata Installation** and notice the search query box contents. For example, for a Host Collection called *my-collection*, the search box contains **host_collection="my-collection"**.

Applying errata to a host

This example uses the API URL for bulk actions **/katello/api/hosts/bulk/install_content** to show the format required for a simple search.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request PUT \
--user sat_username:sat_password --insecure \
--data '{"organization_id":1,"included":{"search":"'my-host'", "content_type":"'errata'", "content":
["RHBA-2016:1981"]}' \
https://satellite.example.com/api/v2/hosts/bulk/install_content
```

Applying errata to a host collection

In this example, notice the level of escaping required to pass the search string **host_collection="my-collection"** as seen in the Satellite web UI.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request PUT \
--user sat_username:sat_password --insecure \
--data '{"organization_id":1,"included":{"search":"'host_collection=\\\\"my-
collection\\\\""', "content_type":"'errata'", "content":["RHBA-2016:1981"]}' \
https://satellite.example.com/api/v2/hosts/bulk/install_content
```

5.5. USING EXTENDED SEARCHES

You can find search parameters that you can use to build your search queries in the web UI. For more information, see [Building Search Queries](#) in *Administering Red Hat Satellite*.

For example, to search for hosts, complete the following steps:

1. In the Satellite web UI, navigate to **Hosts > All Hosts** and click the **Search** field to display a list of search parameters.
2. Locate the search parameters that you want to use. For this example, locate **os_title** and **model**.
3. Combine the search parameters in your API query as follows:

Example request:

```
$ curl --insecure --user sat_username:sat_password \
```

```
https://satellite.example.com/api/v2/hosts?
search=os_title=\"RedHat+7.7\",model=\"PowerEdge+R330\" \
| python3 -m json.tool
```



NOTE

The example request uses **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Example response:

```
{
  ...
  "results": [
    {
      "model_id": 1,
      "model_name": "PowerEdge R330",
      "name": "satellite.example.com",
      "operatingsystem_id": 1,
      "operatingsystem_name": "RedHat 7.7",
      ...
    }
  ],
  "search": "os_title=\"RedHat 7.7\",model=\"PowerEdge R330\"",
  "subtotal": 1,
  "total": 11
}
```

5.6. USING SEARCHES WITH PAGINATION CONTROL

You can use the **per_page** and **page** pagination parameters to limit the search results that an API search query returns. The **per_page** parameter specifies the number of results per page and the **page** parameter specifies which page, as calculated by the **per_page** parameter, to return.

The default number of items to return is set to 1000 when you do not specify any pagination parameters, but the **per_page** value has a default of 20 which applies when you specify the **page** parameter.

Listing content views

This example returns a list of Content Views in pages. The list contains 10 keys per page and returns the third page.

Example request:

```
$ curl --request GET --user sat_username:sat_password \
https://satellite.example.com/katello/api/content_views?per_page=10&page=3
```

Listing activation keys

This example returns a list of activation keys for an organization with ID **1** in pages. The list contains 30 keys per page and returns the second page.

Example request:

```
$ curl --request GET --user sat_username:sat_password \
https://satellite.example.com/katello/api/activation_keys?
organization_id=1&per_page=30&page=2
```

Returning multiple pages

You can use a **for** loop structure to get multiple pages of results.

This example returns pages 1 to 3 of Content Views with 5 results per page:

```
$ for i in seq 1 3; do \
curl --request GET --user sat_username:sat_password \
https://satellite.example.com/katello/api/content_views?per_page=5&page=$i; \
done
```

5.7. OVERRIDING SMART CLASS PARAMETERS

You can search for Smart Parameters using the API and supply a value to override a Smart Parameter in a Class. You can find the full list of attributes that you can modify in the built-in API reference at https://satellite.example.com/apidoc/v2/smart_class_parameters/update.html.

1. Find the ID of the Smart Class parameter you want to change:

- List all Smart Class Parameters.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/smart_class_parameters
```

- If you know the Puppet class ID, for example 5, you can restrict the scope:

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/puppetclasses/5/smart_class_parameters
```

Both calls accept a search parameter. You can view the full list of searchable fields in the Satellite web UI. Navigate to **Configure** > **Smart variables** and click in the search query box to reveal the list of fields.

Two particularly useful search parameters are **puppetclass_name** and **key**, which you can use to search for a specific parameter. For example, using the **--data** option to pass URL encoded data.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
--data 'search=puppetclass_name = access_insights_client and key = authmethod' \
https://satellite.example.com/api/smart_class_parameters
```

Satellite supports standard scoped-search syntax.

2. When you find the ID of the parameter, list the full details including current override values.

Example request:

```
$ curl --request GET --insecure --user sat_username:sat_password \
https://satellite.example.com/api/smart_class_parameters/63
```

3. Enable overriding of parameter values.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request PUT --insecure --user sat_username:sat_password \
--data '{"smart_class_parameter":{"override":true}}' \
https://satellite.example.com/api/smart_class_parameters/63
```

Note that you cannot create or delete the parameters manually. You can only modify their attributes. Satellite creates and deletes parameters only upon class import from a proxy.

4. Add custom override matchers.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request PUT --insecure --user sat_username:sat_password \
--data '{"smart_class_parameter":{"override_value":
{"match":{"hostgroup=Test","value":"2.4.6"}}}' \
https://satellite.example.com/api/smart_class_parameters/63
```

For more information about override values, see https://satellite.example.com/apidoc/v2/override_values.html.

5. You can delete override values.

Example request:

```
$ curl --request DELETE --user sat_username:sat_password \
https://satellite.example.com/api/smart_class_parameters/63/override_values/3
```

5.8. MODIFYING A SMART CLASS PARAMETER USING AN EXTERNAL FILE

Using external files simplifies working with JSON data. Using an editor with syntax highlighting can help you avoid and locate mistakes.



NOTE

The example requests below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Modifying a Smart Class parameter using an external file

This example uses a MOTD Puppet manifest.

1. Search for the Puppet Class by name, **motd** in this case.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" \
--request GET --user sat_user:sat_password --insecure \
https://satellite.example.com/api/smart_class_parameters?search=puppetclass_name=motd \
| python3 -m json.tool
```

- Examine the following output. Each Smart Class Parameter has an ID that is global for the same Satellite instance. The **content** parameter of the **motd** class has **id=3** in this Satellite Server. Do not confuse this with the Puppet Class ID that displays before the Puppet Class name.

Example response:

```
{
  "avoid_duplicates": false,
  "created_at": "2017-02-06 12:37:48 UTC", # Remove this line.
  "default_value": "", # Add a new value here.
  "description": "",
  "hidden_value": "",
  "hidden_value?": false,
  "id": 3,
  "merge_default": false,
  "merge_overrides": false,
  "override": false, # Set the override value to true.
  "override_value_order": "fqdn\nhostgroup\nos\ndomain",
  "override_values": [], # Remove this line.
  "override_values_count": 0,
  "parameter": "content",
  "parameter_type": "string",
  "puppetclass_id": 3,
  "puppetclass_name": "motd",
  "required": false,
  "updated_at": "2017-02-07 11:56:55 UTC", # Remove this line.
  "use_puppet_default": false,
  "validator_rule": null,
  "validator_type": ""
}
```

- Use the parameter ID **3** to get the information specific to the **motd** parameter and redirect the output to a file, for example, **output_file.json**.

Example request:

```
$ curl --header "Accept:application/json" \
--header "Content-Type:application/json" --request GET \
--user sat_user:sat_password --insecure \
https://satellite.example.com/api/smart_class_parameters/3 \
| python3 -m json.tool > output_file.json
```

- Copy the file created in the previous step to a new file for editing, for example, **changed_file.json**:

```
$ cp output_file.json changed_file.json
```

5. Modify the required values in the file. In this example, change the content parameter of the **motd** module, which requires changing the **override** option from **false** to **true**:

```
{
  "avoid_duplicates": false,
  "created_at": "2017-02-06 12:37:48 UTC", # Remove this line.
  "default_value": "", # Add a new value here.
  "description": "",
  "hidden_value": "",
  "hidden_value?": false,
  "id": 3,
  "merge_default": false,
  "merge_overrides": false,
  "override": false, # Set the override value to true.
  "override_value_order": "fqdn\nhostgroup\nos\ndomain",
  "override_values": [], # Remove this line.
  "override_values_count": 0,
  "parameter": "content",
  "parameter_type": "string",
  "puppetclass_id": 3,
  "puppetclass_name": "motd",
  "required": false,
  "updated_at": "2017-02-07 11:56:55 UTC", # Remove this line.
  "use_puppet_default": false,
  "validator_rule": null,
  "validator_type": ""
}
```

6. After editing the file, verify that it looks as follows and then save the changes:

```
{
  "avoid_duplicates": false,
  "default_value": "No Unauthorized Access Allowed",
  "description": "",
  "hidden_value": "",
  "hidden_value?": false,
  "id": 3,
  "merge_default": false,
  "merge_overrides": false,
  "override": true,
  "override_value_order": "fqdn\nhostgroup\nos\ndomain",
  "override_values_count": 0,
  "parameter": "content",
  "parameter_type": "string",
  "puppetclass_id": 3,
  "puppetclass_name": "motd",
  "required": false,
  "use_puppet_default": false,
  "validator_rule": null,
  "validator_type": ""
}
```

7. Apply the changes to Satellite Server:

```
$ curl --header "Accept:application/json" \
```

```
--header "Content-Type:application/json" \
--request PUT --user sat_username:sat_password --insecure \
--data @changed_file.json \
https://satellite.example.com/api/smart_class_parameters/3
```

5.9. DELETING OPENSAP REPORTS

In Satellite Server, you can delete one or more OpenSCAP reports. However, when you delete reports, you must delete one page at a time. If you want to delete all Openscap reports, use the bash script that follows.



NOTE

The example request and the example script below use **python3** to format the response from the Satellite Server. On RHEL 7 and some older systems, you must use **python** instead of **python3**.

Deleting an OpenSCAP report

To delete an OpenSCAP report, complete the following steps:

1. List all OpenSCAP reports. Note the IDs of the reports that you want to delete.

Example request:

```
curl --insecure --user username:_password_ \
https://satellite.example.com/api/v2/compliance/arf_reports/ | python3 -m json.tool
```

Example response:

```
% Total  % Received % Xferd  Average Speed  Time  Time  Time  Current
                        Dload Upload  Total  Spent  Left  Speed
100 3252  0 3252  0  0 4319   0 --:--:-- --:--:-- --:--:-- 4318
{
  "page": 1,
  "per_page": 20,
  "results": [
    {
      "created_at": "2017-05-16 13:27:09 UTC",
      "failed": 0,
      "host": "host1.example.com",
      "id": 404,
      "othered": 0,
      "passed": 0,
      "updated_at": "2017-05-16 13:27:09 UTC"
    },
    {
      "created_at": "2017-05-16 13:26:07 UTC",
      "failed": 0,
      "host": "host2.example.com",
      "id": 405,
      "othered": 0,
      "passed": 0,
      "updated_at": "2017-05-16 13:26:07 UTC"
    }
  ],
}
```

```
{
  "created_at": "2017-05-16 13:25:07 UTC",
  "failed": 0,
  "host": "host3.example.com",
  "id": 406,
  "othered": 0,
  "passed": 0,
  "updated_at": "2017-05-16 13:25:07 UTC"
},
{
  "created_at": "2017-05-16 13:24:07 UTC",
  "failed": 0,
  "host": "host4.example.com",
  "id": 407,
  "othered": 0,
  "passed": 0,
  "updated_at": "2017-05-16 13:24:07 UTC"
},
],
"search": null,
"sort": {
  "by": null,
  "order": null
},
"subtotal": 29,
"total": 29
```

- Using an ID from the previous step, delete the OpenSCAP report. Repeat for each ID that you want to delete.

Example request:

```
# curl --insecure --user username:_password_ \
--header "Content-Type: application/json" \
--request DELETE https://satellite.example.com/api/v2/compliance/arf_reports/405
```

Example response:

```
HTTP/1.1 200 OK
Date: Thu, 18 May 2017 07:14:36 GMT
Server: Apache/2.4.6 (Red Hat Enterprise Linux)
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Foreman_version: 1.11.0.76
Foreman_api_version: 2
Apiipie-Checksum: 2d39dc59aed19120d2359f7515e10d76
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: f47eb877-35c7-41fe-b866-34274b56c506
X-Runtime: 0.661831
X-Powered-By: Phusion Passenger 4.0.18
Set-Cookie: request_method=DELETE; path=/
Set-Cookie: _session_id=d58fe2649e6788b87f46eabf8a461edd; path=/; secure; HttpOnly
ETag: "2574955fc0afc47cb5394ce95553f428"
Status: 200 OK
```

```
Vary: Accept-Encoding
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
```

Example BASH script to delete all OpenSCAP reports

Use the following bash script to delete all the OpenSCAP reports:

```
#!/bin/bash

#this script removes all the arf reports from the satellite server

#settings
USER=username
PASS=password
URI=https://satellite.example.com

#check amount of reports
while [ $(curl --insecure --user $USER:$PASS $URI/api/v2/compliance/arf_reports/ | python3 -m
json.tool | grep "\"total\": | cut --fields=2 --delimiter\":\" | cut --fields=1 --delimiter\", | sed "s/ //g") -gt 0 ];
do

#fetch reports
for i in $(curl --insecure --user $USER:$PASS $URI/api/v2/compliance/arf_reports/ | python3 -m
json.tool | grep "\"id\": | cut --fields=2 --delimiter\":\" | cut --fields=1 --delimiter\", | sed "s/ //g")

#delete reports
do
curl --insecure --user $USER:$PASS --header "Content-Type: application/json" --request DELETE
$URI/api/v2/compliance/arf_reports/$i
done
done
```

5.10. WORKING WITH PULP USING SATELLITE API

When sending API requests to Pulp integrated with Satellite, use certificate-based authentication.

The following examples of Pulp API requests include examples of how to use the Pulp CLI as an alternative. When you run **pulp** commands as root, Pulp CLI uses system certificates configured in **/root/.config/pulp/cli.toml**.

Listing repositories

The endpoint to list all repositories is **/pulp/api/v3/repositories/**. The following query obtains a list of repositories from *satellite.example.com* while supplying the certificates necessary to issue a request from a Satellite Server.

Example request:

```
curl --cacert /etc/pki/katello/certs/katello-server-ca.crt \
--cert /etc/foreman/client_cert.pem --key /etc/foreman/client_key.pem \
https://<satellite.example.com>/pulp/api/v3/repositories/ \
| python3 -m json.tool
```

Example response:

```
{
  "count": 1,
  "next": null,
  "previous": null,
  "results": [
    {
      "pulp_href": "/pulp/api/v3/repositories/rpm/rpm/018cd05a-4b83-73db-b71c-587c6181d89b/",
      "pulp_created": "2024-01-03T17:23:47.715882Z",
      "versions_href": "/pulp/api/v3/repositories/rpm/rpm/018cd05a-4b83-73db-b71c-587c6181d89b/versions/",
      "pulp_labels": {},
      "latest_version_href": "/pulp/api/v3/repositories/rpm/rpm/018cd05a-4b83-73db-b71c-587c6181d89b/versions/1/",
      "name": "Red_Hat_Enterprise_Linux_8_for_x86_64_-_BaseOS_Kickstart_8_9-49838",
      "description": null,
      "retain_repo_versions": null,
      "remote": null
    }
  ]
}
```

Alternatively, use the Pulp CLI to list repositories:

```
# pulp repository list
[
  {
    "pulp_href": "/pulp/api/v3/repositories/rpm/rpm/018cd025-c6ef-7237-a99e-70bab3d30941/",
    "pulp_created": "2024-01-03T16:26:25.904682Z",
    "versions_href": "/pulp/api/v3/repositories/rpm/rpm/018cd025-c6ef-7237-a99e-70bab3d30941/versions/",
    "pulp_labels": {},
    "latest_version_href": "/pulp/api/v3/repositories/rpm/rpm/018cd025-c6ef-7237-a99e-70bab3d30941/versions/1/",
    "name": "Red_Hat_Enterprise_Linux_8_for_x86_64_-_AppStream_RPMs_8-2875",
    "description": null,
    "retain_repo_versions": null,
    "remote": null
  }
]
```

Inspecting Pulp status

The endpoint to return status information about Pulp is **/pulp/api/v3/status/**. Requests for Pulp Status do not require authentication.

Example request:

```
curl https://<satellite.example.com>/pulp/api/v3/status/ \
| python3 -m json.tool
```

Example response:

```
{
  "versions": [
    {
```

```
    "component": "core",
    "version": "3.39.4",
    "package": "pulpcore",
    "domain_compatible": true
  },
  {
    "component": "rpm",
    "version": "3.23.0",
    "package": "pulp-rpm",
    "domain_compatible": true
  },
  ...
```

Alternatively, use the Pulp CLI to retrieve Pulp status:

```
# pulp status
{
  "versions": [
    {
      "component": "core",
      "version": "3.39.4",
      "package": "pulpcore",
      "domain_compatible": true
    },
    {
      "component": "rpm",
      "version": "3.23.0",
      "package": "pulp-rpm",
      "domain_compatible": true
    },
    ...
```

Additional resources

- Run **pulp --help** for details on how to use the Pulp CLI.
- The full API reference for Pulp is available on your Satellite Server at **<https://<satellite.example.com>/pulp/api/v3/docs/>**.

APPENDIX A. API RESPONSE CODES

The Red Hat Satellite 6 API provides HTTP response status codes for API calls. The following codes are common for all resources in the Satellite API.

Table A.1. API response codes

Response	Explanation
200 OK	For a successful request action: show, index, update, or delete (GET, PUT, DELETE requests).
201 Created	For a successful create action (POST request).
301 Moved Permanently	Redirect when Satellite is restricted to use HTTPS and HTTP is attempted.
400 Bad Request	A required parameter is missing or the search query has invalid syntax.
401 Unauthorized	Failed to authorize the user (for example, incorrect credentials).
403 Forbidden	The user has insufficient permissions to perform the action or read the resource, or the action is unsupported in general.
404 Not Found	The record with the given ID does not exist. It can appear in show and delete actions when the requested record does not exist; or in create, update and delete actions when one of the associated records does not exist.
409 Conflict	Could not delete the record due to existing dependencies (for example, host groups with hosts).
415 Unsupported Media Type	The content type of the HTTP request is not JSON.
422 Unprocessable Entity	Failed to create an entity due to some validation errors. Applies to create or update actions only.
500 Internal Server Error	Unexpected internal server error.
503 Service Unavailable	The server is not running.

APPENDIX B. API PERMISSIONS MATRIX

The Red Hat Satellite 6 API supports numerous actions, many of which require specific permissions. The following table lists the API permission names, the actions associated with those permissions, and the associated resource type.

Table B.1. API permissions matrix

Permission Name	Actions	Resource Type
view_activation_keys	- katello/activation_keys/all - katello/activation_keys/index - katello/activation_keys/auto_complete_search - katello/api/v2/activation_keys/index - katello/api/v2/activation_keys/show - katello/api/v2/activation_keys/available_host_collections - katello/api/v2/activation_keys/available_releases - katello/api/v2/activation_keys/product_content	Katello::ActivationKey
create_activation_keys	- katello/api/v2/activation_keys/create - katello/api/v2/activation_keys/copy	Katello::ActivationKey
edit_activation_keys	- katello/api/v2/activation_keys/update - katello/api/v2/activation_keys/content_override - katello/api/v2/activation_keys/add_subscriptions - katello/api/v2/activation_keys/remove_subscriptions	Katello::ActivationKey

Permission Name	Actions	Resource Type
destroy_activation_keys	katello/api/v2/activation_keys/destroy	Katello::ActivationKey
logout	users/logout	
view_architectures	- architectures/index - architectures/show - architectures/auto_complete_search - api/v2/architectures/index - api/v2/architectures/show	
create_architectures	- architectures/new - architectures/create - api/v2/architectures/create	
edit_architectures	- architectures/edit - architectures/update - api/v2/architectures/update	
destroy_architectures	- architectures/destroy - api/v2/architectures/destroy	

Permission Name	Actions	Resource Type
view_audit_logs	- audits/index - audits/show - audits/auto_complete_search - api/v2/audits/index - api/v2/audits/show	
view_authenticators	- auth_source_ldaps/index - auth_source_ldaps/show - api/v2/auth_source_ldaps/index - api/v2/auth_source_ldaps/show	
create_authenticators	- auth_source_ldaps/new - auth_source_ldaps/create - api/v2/auth_source_ldaps/create	
edit_authenticators	- auth_source_ldaps/edit - auth_source_ldaps/update - api/v2/auth_source_ldaps/update	
destroy_authenticators	- auth_source_ldaps/destroy - api/v2/auth_source_ldaps/destroy	

Permission Name	Actions	Resource Type
view_bookmarks	● bookmarks/index ● bookmarks/show ● api/v2/bookmarks/index ● api/v2/bookmarks/show	
create_bookmarks	● bookmarks/new ● bookmarks/create ● api/v2/bookmarks/new ● api/v2/bookmarks/create	
edit_bookmarks	● bookmarks/edit ● bookmarks/update ● api/v2/bookmarks/edit ● api/v2/bookmarks/update	
destroy_bookmarks	● bookmarks/destroy ● api/v2/bookmarks/destroy	

Permission Name	Actions	Resource Type
download_bootdisk	● foreman_bootdisk/disks/generic ● foreman_bootdisk/disks/host ● foreman_bootdisk/disks/full_host ● foreman_bootdisk/disks/subnet ● foreman_bootdisk/disks/help ● foreman_bootdisk/api/v2/disks/generic ● foreman_bootdisk/api/v2/disks/host	
manage_capsule_content	● katello/api/v2/capsule_content/lifecycle_environments ● katello/api/v2/capsule_content/available_lifecycle_environments ● katello/api/v2/capsule_content/add_lifecycle_environment ● katello/api/v2/capsule_content/remove_lifecycle_environment ● katello/api/v2/capsule_content/sync ● katello/api/v2/capsule_content/sync_status ● katello/api/v2/capsule_content/cancel_sync	SmartProxy
view_capsule_content	● smart_proxies/pulp_storage ● smart_proxies/pulp_status ● smart_proxies/show_with_content	SmartProxy

Permission Name	Actions	Resource Type
view_compute_profiles	• compute_profiles/index • compute_profiles/show • compute_profiles/auto_complete_search • api/v2/compute_profiles/index • api/v2/compute_profiles/show	
create_compute_profiles	• compute_profiles/new • compute_profiles/create • api/v2/compute_profiles/create	
edit_compute_profiles	• compute_profiles/edit • compute_profiles/update • api/v2/compute_profiles/update	
destroy_compute_profiles	• compute_profiles/destroy • api/v2/compute_profiles/destroy	

Permission Name	Actions	Resource Type
view_compute_resources	● <code>compute_resources/index</code> ● <code>compute_resources/show</code> ● <code>compute_resources/auto_complete_search</code> ● <code>compute_resources/ping</code> ● <code>compute_resources/available_images</code> ● <code>api/v2/compute_resources/index</code> ● <code>api/v2/compute_resources/show</code> ● <code>api/v2/compute_resources/available_images</code> ● <code>api/v2/compute_resources/available_clusters</code> ● <code>api/v2/compute_resources/available_folders</code> ● <code>api/v2/compute_resources/available_flavors</code> ● <code>api/v2/compute_resources/available_networks</code> ● <code>api/v2/compute_resources/available_resource_pools</code> ● <code>api/v2/compute_resources/available_security_groups</code> ● <code>api/v2/compute_resources/available_storage_domains</code> ● <code>api/v2/compute_resources/available_zones</code> ● <code>api/v2/compute_resources/available_storage_pods</code>	

Permission Name	Actions	Resource Type
create_compute_resources	● compute_resources/new ● compute_resources/create ● compute_resources/test_connection ● api/v2/compute_resources/create	
edit_compute_resources	● compute_resources/edit ● compute_resources/update ● compute_resources/test_connection ● compute_attributes/new ● compute_attributes/create ● compute_attributes/edit ● compute_attributes/update ● api/v2/compute_resources/update ● api/v2/compute_attributes/create ● api/v2/compute_attributes/update	
destroy_compute_resources	● compute_resources/destroy ● api/v2/compute_resources/destroy	
view_compute_resources_vms	● compute_resources_vms/index ● compute_resources_vms/show	

Permission Name	Actions	Resource Type
create_compute_resources_vms	• <code>compute_resources_vms/new</code> • <code>compute_resources_vms/create</code>	
edit_compute_resources_vms	• <code>compute_resources_vms/edit</code> • <code>compute_resources_vms/update</code>	
destroy_compute_resources_vms	• <code>compute_resources_vms/destroy</code>	
power_compute_resources_vms	• <code>compute_resources_vms/power</code> • <code>compute_resources_vms/pause</code>	
console_compute_resources_vms	• <code>compute_resources_vms/console</code>	
view_config_groups	• <code>config_groups/index</code> • <code>config_groups/auto_complete_search</code> • <code>api/v2/config_groups/index</code> • <code>api/v2/config_groups/show</code>	
create_config_groups	• <code>config_groups/new</code> • <code>config_groups/create</code> • <code>api/v2/config_groups/create</code>	

Permission Name	Actions	Resource Type
edit_config_groups	● config_groups/edit ● config_groups/update ● api/v2/config_groups/update	
destroy_config_groups	● config_groups/destroy ● api/v2/config_groups/destroy	
view_config_reports	● config_reports/index ● config_reports/show ● config_reports/autocomplete_search ● api/v2/config_reports/index ● api/v2/config_reports/show ● api/v2/config_reports/latest	
destroy_config_reports	● config_reports/destroy ● api/v2/config_reports/destroy	
upload_config_reports	● api/v2/config_reports/create	
view_containers	● containers/index ● containers/show ● api/v2/containers/index ● api/v2/containers/show ● api/v2/containers/logs	Container

Permission Name	Actions	Resource Type
commit_containers	containers/commit	Container
create_containers	- containers/steps/show - containers/steps/update - containers/new - api/v2/containers/create - api/v2/containers/power	Container
destroy_containers	- containers/destroy - api/v2/containers/destroy	Container
power_compute_resources_vms	- containers/power - api/v2/containers/create - api/v2/containers/power	ComputeResource
view_content_views	- katello/api/v2/content_views/index - katello/api/v2/content_views/show - katello/api/v2/content_views/available_puppet_modules - katello/api/v2/content_views/available_puppet_module_names - katello/api/v2/content_view_filters/index - katello/api/v2/content_view_filters/show - katello/api/v2/content_view_filter_rules/index - katello/api/v2/content_view_filter_rules/show	Katello::ContentView

Permission Name	Actions	Resource Type
	- katello/api/v2/content_views_histories/index - katello/api/v2/content_views_puppet_modules/index - katello/api/v2/content_views_puppet_modules/show - katello/api/v2/content_views_versions/index - katello/api/v2/content_views_versions/show - katello/api/v2/package_groups/index - katello/api/v2/package_groups/show - katello/api/v2/errata/index - katello/api/v2/errata/show - katello/api/v2/puppet_modules/index - katello/api/v2/puppet_modules/show - katello/content_views/automatic_complete - katello/content_views/automatic_complete_search - katello/errata/short_details - katello/errata/automatic_complete - katello/packages/details - katello/packages/automatic_complete - katello/products/automatic_complete - katello/repositories/automatic_complete_library - katello/content_search/index - katello/content_search/products - katello/content_search/repos	

Permission Name	katello/content_search/packages Actions	Resource Type
	- katello/content_search/errata - katello/content_search/puppet_modules - katello/content_search/packages_items - katello/content_search/errata_items - katello/content_search/puppet_modules_items - katello/content_search/view_packages - katello/content_search/view_puppet_modules - katello/content_search/rpo_packages - katello/content_search/rpo_errata - katello/content_search/rpo_puppet_modules - katello/content_search/rpo_compare_errata - katello/content_search/rpo_compare_packages - katello/content_search/rpo_compare_puppet_modules - katello/content_search/view_compare_errata - katello/content_search/view_compare_packages - katello/content_search/view_compare_puppet_modules - katello/content_search/views	
create_content_views	- katello/api/v2/content_views/create - katello/api/v2/content_views/copy	Katello::ContentView

Permission Name	Actions	Resource Type
edit_content_views	- katello/api/v2/content_views/update - katello/api/v2/content_view_filters/create - katello/api/v2/content_view_filters/update - katello/api/v2/content_view_filters/destroy - katello/api/v2/content_view_filter_rules/create - katello/api/v2/content_view_filter_rules/update - katello/api/v2/content_view_filter_rules/destroy - katello/api/v2/content_view_puppet_modules/create - katello/api/v2/content_view_puppet_modules/update - katello/api/v2/content_view_puppet_modules/destroy	Katello::ContentView
destroy_content_views	- katello/api/v2/content_views/destroy - katello/api/v2/content_views/remove - katello/api/v2/content_view_versions/destroy	Katello::ContentView

Permission Name	Actions	Resource Type
publish_content_views	- katello/api/v2/content_views/publish - katello/api/v2/content_view_versions/incremental_update	Katello::ContentView
promote_or_remove_content_views	- katello/api/v2/content_view_versions/promote - katello/api/v2/content_views/remove_from_environment - katello/api/v2/content_views/remove	Katello::ContentView
export_content_views	katello/api/v2/content_view_versions/export	Katello::ContentView
access_dashboard	- dashboard/index - dashboard/save_positions - dashboard/reset_default - dashboard/create - dashboard/destroy - api/v2/dashboard/index	
view_discovered_hosts	- discovered_hosts/index - discovered_hosts/show - discovered_hosts/auto_complete_search - api/v2/discovered_hosts/show	Host

Permission Name	Actions	Resource Type
submit_discovered_hosts	● api/v2/discovered_hosts/facts ● api/v2/discovered_hosts/create	Host
auto_provision_discovered_hosts	● discovered_hosts/auto_provision ● discovered_hosts/auto_provision_all ● api/v2/discovered_hosts/auto_provision ● api/v2/discovered_hosts/auto_provision_all	Host
provision_discovered_hosts	● discovered_hosts/edit ● discovered_hosts/update ● api/v2/discovered_hosts/update	Host

Permission Name	Actions	Resource Type
edit_discovered_hosts	- discovered_hosts/update_multiple_location - discovered_hosts/select_multiple_organization - discovered_hosts/update_multiple_organization - discovered_hosts/select_multiple_location - discovered_hosts/refresh_facts - discovered_hosts/reboot - discovered_hosts/reboot_all - api/v2/discovered_hosts/refresh_facts - api/v2/discovered_hosts/reboot - api/v2/discovered_hosts/reboot_all	Host
destroy_discovered_hosts	- discovered_hosts/destroy - discovered_hosts/submit_multiple_destroy - discovered_hosts/multiple_destroy - api/v2/discovered_hosts/destroy	Host
view_discovery_rules	- discovery_rules/index - discovery_rules/show - discovery_rules/autocomplete_search - api/v2/discovery_rules/index - api/v2/discovery_rules/show	DiscoveryRule

Permission Name	Actions	Resource Type
create_discovery_rules	- discovery_rules/new - discovery_rules/create - api/v2/discovery_rules/create	DiscoveryRule
edit_discovery_rules	- discovery_rules/edit - discovery_rules/update - discovery_rules/enable - discovery_rules/disable - api/v2/discovery_rules/create - api/v2/discovery_rules/update	DiscoveryRule
execute_discovery_rules	- discovery_rules/auto_provision - discovery_rules/auto_provision_all - api/v2/discovery_rules/auto_provision - api/v2/discovery_rules/auto_provision_all	DiscoveryRule
destroy_discovery_rules	- discovery_rules/destroy - api/v2/discovery_rules/destroy	DiscoveryRule

Permission Name	Actions	Resource Type
view_domains	- domains/index - domains/show - domains/auto_complete_search - api/v2/domains/index - api/v2/domains/show - api/v2/parameters/index - api/v2/parameters/show	
create_domains	- domains/new - domains/create - api/v2/domains/create	
edit_domains	- domains/edit - domains/update - api/v2/domains/update - api/v2/parameters/create - api/v2/parameters/update - api/v2/parameters/destroy - api/v2/parameters/reset	
destroy_domains	- domains/destroy - api/v2/domains/destroy	

Permission Name	Actions	Resource Type
view_environments	- environments/index - environments/show - environments/auto_complete_search - api/v2/environments/index - api/v2/environments/show	
create_environments	- environments/new - environments/create - api/v2/environments/create	
edit_environments	- environments/edit - environments/update - api/v2/environments/update	
destroy_environments	- environments/destroy - api/v2/environments/destroy	
import_environments	- environments/import_environments - environments/obsolete_and_new - api/v2/environments/import_puppetclasses - api/v2/smart_proxies/import_puppetclasses	

Permission Name	Actions	Resource Type
view_external_usergroups	- external_usergroups/index - external_usergroups/show - api/v2/external_usergroups/index - api/v2/external_usergroups/show	
create_external_usergroups	- external_usergroups/new - external_usergroups/create - api/v2/external_usergroups/new - api/v2/external_usergroups/create	
edit_external_usergroups	- external_usergroups/edit - external_usergroups/update - external_usergroups/refresh - api/v2/external_usergroups/update - api/v2/external_usergroups/refresh	
destroy_external_usergroups	- external_usergroups/destroy - api/v2/external_usergroups/destroy	

Permission Name	Actions	Resource Type
view_external_variables	● lookup_keys/index ● lookup_keys/show ● lookup_keys/auto_complete_search ● puppetclass_lookup_keys/index ● puppetclass_lookup_keys/show ● puppetclass_lookup_keys/auto_complete_search ● variable_lookup_keys/index ● variable_lookup_keys/show ● variable_lookup_keys/auto_complete_search ● lookup_values/index ● api/v2/smart_variables/index ● api/v2/smart_variables/show ● api/v2/smart_class_parameters/index ● api/v2/smart_class_parameters/show ● api/v2/override_values/index ● api/v2/override_values/show	

Permission Name	Actions	Resource Type
create_external_variables	● lookup_keys/new ● lookup_keys/create ● puppetclass_lookup_keys/new ● puppetclass_lookup_keys/create ● variable_lookup_keys/new ● variable_lookup_keys/create ● lookup_values/create ● api/v2/smart_variables/create ● api/v2/smart_class_parameters/create ● api/v2/override_values/create	

Permission Name	Actions	Resource Type
edit_external_variables	● lookup_keys/edit ● lookup_keys/update ● puppetclass_lookup_keys/edit ● puppetclass_lookup_keys/update ● variable_lookup_keys/edit ● variable_lookup_keys/update ● lookup_values/create ● lookup_values/update ● lookup_values/destroy ● api/v2/smart_variables/update ● api/v2/smart_class_parameters/update ● api/v2/override_values/create ● api/v2/override_values/update ● api/v2/override_values/destroy	

Permission Name	Actions	Resource Type
destroy_external_variables	- lookup_keys/destroy - puppetclass_lookup_keys/destroy - variable_lookup_keys/destroy - lookup_values/destroy - api/v2/smart_variables/destroy - api/v2/smart_class_parameters/destroy - api/v2/override_values/create - api/v2/override_values/update - api/v2/override_values/destroy	
view_facts	- facts/index - facts/show - fact_values/index - fact_values/show - fact_values/auto_complete_search - api/v2/fact_values/index - api/v2/fact_values/show	
upload_facts	api/v2/hosts/facts	
view_filters	- filters/index - filters/auto_complete_search - api/v2/filters/index - api/v2/filters/show	

Permission Name	Actions	Resource Type
create_filters	- filters/new - filters/create - api/v2/filters/create	
edit_filters	- filters/edit - filters/update - permissions/index - api/v2/filters/update - api/v2/permissions/index - api/v2/permissions/show	
destroy_filters	- filters/destroy - api/v2/filters/destroy	
view_arf_reports	- arf_reports/index - arf_reports/show - arf_reports/parse_html - arf_reports/show_html - arf_reports/parse_bzip - arf_reports/auto_complete_search - api/v2/compliance/arf_reports/index - api/v2/compliance/arf_reports/show - compliance_hosts/show	

Permission Name	Actions	Resource Type
destroy_arf_reports	- arf_reports/destroy - arf_reports/delete_multiple - arf_reports/submit_delete_multiple - api/v2/compliance/arf_reports/destroy	
create_arf_reports	api/v2/compliance/arf_reports/create	
view_policies	- policies/index - policies/show - policies/parse - policies/auto_complete_search - policy_dashboard/index - compliance_dashboard/index - api/v2/compliance/policies/index - api/v2/compliance/policies/show - api/v2/compliance/policies/content	ForemanOpenscap::Policy
edit_policies	- policies/edit - policies/update - policies/scap_content_selected - api/v2/compliance/policies/update	ForemanOpenscap::Policy

Permission Name	Actions	Resource Type
create_policies	• policies/new • policies/create • api/v2/compliance/policies/create	ForemanOpenscap::Policy
destroy_policies	• policies/destroy • api/v2/compliance/policies/destroy	ForemanOpenscap::Policy
assign_policies	• policies/select_multiple_hosts • policies/update_multiple_hosts • policies/disassociate_multiple_hosts • policies/remove_policy_from_multiple_hosts	ForemanOpenscap::Policy
view_scap_contents	• scap_contents/index • scap_contents/show • scap_contents/auto_complete_search • api/v2/compliance/scap_contents/index • api/v2/compliance/scap_contents/show	ForemanOpenscap::ScapContent
view_scap_contents	• scap_contents/index • scap_contents/show • scap_contents/auto_complete_search • api/v2/compliance/scap_contents/index • api/v2/compliance/scap_contents/show	ForemanOpenscap::ScapContent

Permission Name	Actions	Resource Type
edit_scap_contents	- scap_contents/edit - scap_contents/update - api/v2/compliance/scap_contents/update	ForemanOpenscap::ScapContent
create_scap_contents	- scap_contents/new - scap_contents/create - api/v2/compliance/scap_contents/create	ForemanOpenscap::ScapContent
destroy_scap_contents	- scap_contents/destroy - api/v2/compliance/scap_contents/destroy	ForemanOpenscap::ScapContent
edit_hosts	hosts/openscap_proxy_changed	Host
edit_hostgroups	hostgroups/openscap_proxy_changed	Host

Permission Name	Actions	Resource Type
view_job_templates	● job_templates/index ● job_templates/show ● job_templates/revision ● job_templates/auto_complete_search ● job_templates/auto_complete_job_category ● job_templates/preview ● job_templates/export ● api/v2/job_templates/index ● api/v2/job_templates/show ● api/v2/job_templates/revision ● api/v2/job_templates/export ● api/v2/template_inputs/index ● api/v2/template_inputs/show ● api/v2/foreign_input_sets/index ● api/v2/foreign_input_sets/show	JobTemplate
create_job_templates	● job_templates/new ● job_templates/create ● job_templates/clone_template ● job_templates/import ● api/v2/job_templates/create ● api/v2/job_templates/clone ● api/v2/job_templates/import	JobTemplate

Permission Name	Actions	Resource Type
edit_job_templates	● job_templates/edit ● job_templates/update ● api/v2/job_templates/update ● api/v2/template_inputs/create ● api/v2/template_inputs/update ● api/v2/template_inputs/destroy ● api/v2/foreign_input_sets/create ● api/v2/foreign_input_sets/update ● api/v2/foreign_input_sets/destroy	
edit_job_templates	● job_templates/edit ● job_templates/update ● api/v2/job_templates/update ● api/v2/template_inputs/create ● api/v2/template_inputs/update ● api/v2/template_inputs/destroy ● api/v2/foreign_input_sets/create ● api/v2/foreign_input_sets/update ● api/v2/foreign_input_sets/destroy	

Permission Name	Actions	Resource Type
edit_remote_execution_features	- remote_execution_features/index - remote_execution_features/show - remote_execution_features/update - api/v2/remote_execution_features/index - api/v2/remote_execution_features/show - api/v2/remote_execution_features/update	RemoteExecutionFeature
destroy_job_templates	- job_templates/destroy - api/v2/job_templates/destroy	JobTemplate
lock_job_templates	- job_templates/lock - job_templates/unlock	JobTemplate
create_job_invocations	- job_invocations/new - job_invocations/create - job_invocations/refresh - job_invocations/rerun - job_invocations/preview_hosts - api/v2/job_invocations/create	JobInvocation

Permission Name	Actions	Resource Type
view_job_invocations	• job_invocations/index • job_invocations/show • template_invocations/show • api/v2/job_invocations/index • api/v2/job_invocations/show • api/v2/job_invocations/output	JobInvocation
execute_template_invocation		TemplateInvocation
filter_autocompletion_for_template_invocation	• template_invocations/autocomplete_search • job_invocations/show • template_invocations/index	TemplateInvocation
view_foreman_tasks	• foreman_tasks/tasks/autocomplete_search • foreman_tasks/tasks/sub_tasks • foreman_tasks/tasks/index • foreman_tasks/tasks/show • foreman_tasks/api/tasks/bulk_search • foreman_tasks/api/tasks/show • foreman_tasks/api/tasks/index • foreman_tasks/api/tasks/summary	ForemanTasks::Task

Permission Name	Actions	Resource Type
edit_foreman_tasks	● foreman_tasks/tasks/resume ● foreman_tasks/tasks/unlock ● foreman_tasks/tasks/force_unlock ● foreman_tasks/tasks/cancel_step ● foreman_tasks/api/tasks/bulk_resume	ForemanTasks::Task
create_recurring_logics		ForemanTasks::RecurringLogic
view_recurring_logics	● foreman_tasks/recurring_logics/index ● foreman_tasks/recurring_logics/show ● foreman_tasks/api/recurring_logics/index ● foreman_tasks/api/recurring_logics/show	ForemanTasks::RecurringLogic
edit_recurring_logics	● foreman_tasks/recurring_logics/cancel ● foreman_tasks/api/recurring_logics/cancel	ForemanTasks::RecurringLogic
view_globals	● common_parameters/index ● common_parameters/show ● common_parameters/autocomplete_search ● api/v2/common_parameters/index ● api/v2/common_parameters/show	

Permission Name	Actions	Resource Type
create_globals	- common_parameters/new - common_parameters/create - api/v2/common_parameters/create	
edit_globals	- common_parameters/edit - common_parameters/update - api/v2/common_parameters/update	
destroy_globals	- common_parameters/destroy - api/v2/common_parameters/destroy	
view_gpg_keys	- katello/gpg_keys/all - katello/gpg_keys/index - katello/gpg_keys/auto_complete_search - katello/api/v2/gpg_keys/index - katello/api/v2/gpg_keys/show	Katello::GpgKey
create_gpg_keys	katello/api/v2/gpg_keys/create	Katello::GpgKey
edit_gpg_keys	- katello/api/v2/gpg_keys/update - katello/api/v2/gpg_keys/content	Katello::GpgKey

Permission Name	Actions	Resource Type
destroy_gpg_keys	katello/api/v2/gpg_keys/destroy	Katello::GpgKey
view_host_collections	- katello/api/v2/host_collections/index - katello/api/v2/host_collections/show - katello/host_collections/auto_complete_search	Katello::HostCollection
create_host_collections	- katello/api/v2/host_collections/create - katello/api/v2/host_collections/copy	Katello::HostCollection
edit_host_collections	- katello/api/v2/host_collections/update - katello/api/v2/host_collections/add_systems - katello/api/v2/host_collections/remove_systems	Katello::HostCollection
destroy_host_collections	katello/api/v2/host_collections/destroy	Katello::HostCollection
edit_classes	- host_editing/edit_classes - api/v2/host_classes/index - api/v2/host_classes/create - api/v2/host_classes/destroy	

Permission Name	Actions	Resource Type
create_params	● host_editing/create_params ● api/v2/parameters/create	
edit_params	● host_editing/edit_params ● api/v2/parameters/update	
destroy_params	● host_editing/destroy_params ● api/v2/parameters/destroy ● api/v2/parameters/reset	
view_hostgroups	● hostgroups/index ● hostgroups/show ● hostgroups/auto_complete_search ● api/v2/hostgroups/index ● api/v2/hostgroups/show	

Permission Name	Actions	Resource Type
create_hostgroups	● hostgroups/new ● hostgroups/create ● hostgroups/clone ● hostgroups/nest ● hostgroups/process_hostgroup ● hostgroups/architecture_selected ● hostgroups/domain_selected ● hostgroups/environment_selected ● hostgroups/medium_selected ● hostgroups/os_selected ● hostgroups/use_image_selected ● hostgroups/process_hostgroup ● hostgroups/puppetclass_parameters ● host/process_hostgroup ● puppetclasses/parameters ● api/v2/hostgroups/create ● api/v2/hostgroups/clone	

Permission Name	Actions	Resource Type
edit_hostgroups	● hostgroups/edit ● hostgroups/update ● hostgroups/architecture_selected ● hostgroups/process_hostgroup ● hostgroups/architecture_selected ● hostgroups/domain_selected ● hostgroups/environment_selected ● hostgroups/medium_selected ● hostgroups/os_selected ● hostgroups/use_image_selected ● hostgroups/process_hostgroup ● hostgroups/puppetclass_parameters ● host/process_hostgroup ● puppetclasses/parameters ● api/v2/hostgroups/update ● api/v2/parameters/create ● api/v2/parameters/update ● api/v2/parameters/destroy ● api/v2/parameters/reset ● api/v2/hostgroup_classes/index ● api/v2/hostgroup_classes/create ● api/v2/hostgroup_classes/destroy	

Permission Name	Actions	Resource Type
destroy_hostgroups	● hostgroups/destroy ● api/v2/hostgroups/destroy	
view_hosts	● hosts/index ● hosts/show ● hosts/errors ● hosts/active ● hosts/out_of_sync ● hosts/disabled ● hosts/pending ● hosts/vm ● hosts/externalNodes ● hosts/pxe_config ● hosts/storeconfig_klasses ● hosts/auto_complete_search ● hosts/bmc ● hosts/runtime ● hosts/resources ● hosts/templates ● hosts/overview ● hosts/nics ● dashboard/OutOfSync ● dashboard/errors ● dashboard/active ● unattended/host_template ● unattended/hostgroup_template ● api/v2/hosts/index ● api/v2/hosts/show	

Permission Name	Actions	Resource Type
	• api/v2/hosts/status/configuration • api/v2/hosts/get_status • api/v2/hosts/vm_compute_attributes • api/v2/hosts/template • api/v2/interfaces/index • api/v2/interfaces/show • locations/mismatches • organizations/mismatches • hosts/puppet_environment_for_content_view • katello/api/v2/host_auto_complete/auto_complete_search • katello/api/v2/host_errata/index • katello/api/v2/host_errata/show • katello/api/v2/host_errata/auto_complete_search • katello/api/v2/host_subscriptions/index • katello/api/v2/host_subscriptions/events • katello/api/v2/host_subscriptions/product_content • katello/api/v2/hosts_bulk_actions/installable_errata • katello/api/v2/hosts_bulk_actions/available_incremental_updates • katello/api/v2/host_packages/index	
create_hosts	• hosts/new • hosts/create • hosts/clone	

Permission Name	Actions	Resource Type
	• hosts/architecture_selected • hosts/compute_resource_selected • hosts/domain_selected • hosts/environment_selected • hosts/hostgroup_or_environment_selected • hosts/medium_selected • hosts/os_selected • hosts/use_image_selected • hosts/process_hostgroup • hosts/process_taxonomy • hosts/current_parameters • hosts/puppetclass_parameters • hosts/template_used • hosts/interfaces • compute_resources/cluster_selected • compute_resources/template_selected • compute_resources/provider_selected • compute_resources/resource_pools • puppetclasses/parameters • subnets/freeip • interfaces/new • api/v2/hosts/create • api/v2/interfaces/create • api/v2/tasks/index	
edit_hosts	• hosts/edit • hosts/update	

Permission Name	Actions	Resource Type
	• hosts/multiple_actions • hosts/reset_multiple • hosts/submit_multiple_enable • hosts/select_multiple_hostgroup • hosts/select_multiple_environment • hosts/submit_multiple_disable • hosts/multiple_parameters • hosts/multiple_disable • hosts/multiple_enable • hosts/update_multiple_environment • hosts/update_multiple_hostgroup • hosts/update_multiple_parameters • hosts/toggle_manage • hosts/select_multiple_organization • hosts/update_multiple_organization • hosts/disassociate • hosts/multiple_disassociate • hosts/update_multiple_disassociate • hosts/select_multiple_owner • hosts/update_multiple_owner • hosts/select_multiple_power_state • hosts/update_multiple_power_state • hosts/select_multiple_puppet_proxy • hosts/update_multiple_puppet_proxy	

Permission Name	Actions	Resource Type
	• hosts/select_multiple_puppet_ca_proxy • hosts/update_multiple_puppet_ca_proxy • hosts/select_multiple_location • hosts/update_multiple_location • hosts/architecture_selected • hosts/compute_resource_selected • hosts/domain_selected • hosts/environment_selected • hosts/hostgroup_or_environment_selected • hosts/medium_selected • hosts/os_selected • hosts/use_image_selected • hosts/process_hostgroup • hosts/process_taxonomy • hosts/current_parameters • hosts/puppetclass_parameters • hosts/template_used • hosts/interfaces • compute_resources/associate • compute_resources/[:cluster_selected, :template_selected, :provider_selected, :resource_pools] • compute_resources_vms/associate • puppetclasses/parameters • subnets/freeip	

Permission Name	Actions	Resource Type
	• interfaces/new • api/v2/hosts/update • api/v2/hosts/disassociate • api/v2/interfaces/create • api/v2/interfaces/update • api/v2/interfaces/destroy • api/v2/compute_resources/associate • api/v2/hosts/host_collections • katello/api/v2/host_errata/apply • katello/api/v2/host_packages/install • katello/api/v2/host_packages/upgrade • katello/api/v2/host_packages/upgrade_all • katello/api/v2/host_packages/remove • katello/api/v2/host_subscriptions/auto_attach • katello/api/v2/host_subscriptions/add_subscriptions • katello/api/v2/host_subscriptions/remove_subscriptions • katello/api/v2/host_subscriptions/content_override • katello/api/v2/hosts_bulk_actions/bulk_add_host_collections • katello/api/v2/hosts_bulk_actions/bulk_remove_host_collections • katello/api/v2/hosts_bulk_actions/install_content • katello/api/v2/hosts_bulk_actions/update_content	

Permission Name	Actions	Resource Type
	- katello/api/v2/hosts_bulk_actions/remove_content - katello/api/v2/hosts_bulk_actions/environment_content_view	
destroy_hosts	- hosts/destroy - hosts/multiple_actions - hosts/reset_multiple - hosts/multiple_destroy - hosts/submit_multiple_destroy - api/v2/hosts/destroy - api/v2/interfaces/destroy - katello/api/v2/hosts_bulk_actions/destroy_hosts	
build_hosts	- hosts/setBuild - hosts/cancelBuild - hosts/multiple_build - hosts/submit_multiple_build - hosts/review_before_build - hosts/rebuild_config - hosts/submit_rebuild_config - tasks/show - api/v2/tasks/index - api/v2/hosts/rebuild_config	
power_hosts	- hosts/power - api/v2/hosts/power	

Permission Name	Actions	Resource Type
console_hosts	• hosts/console	
ipmi_boot	• hosts/ipmi_boot • api/v2/hosts/boot	
puppetrun_hosts	• hosts/puppetrun • hosts/multiple_puppetrun • hosts/update_multiple_puppetrun • api/v2/hosts/puppetrun	
search_repository_image_search	• image_search/auto_complete_repository_name • image_search/auto_complete_image_tag • image_search/search_repository	Docker/ImageSearch
view_images	• images/index • images/show • images/auto_complete_search • api/v2/images/index • api/v2/images/show	
create_images	• images/new • images/create • api/v2/images/create	

Permission Name	Actions	Resource Type
edit_images	- images/edit - images/update - api/v2/images/update	
destroy_images	- images/destroy - api/v2/images/destroy	
view_lifecycle_environments	- katello/api/v2/environments/index - katello/api/v2/environments/show - katello/api/v2/environments/paths - katello/api/v2/environments/repositories - katello/api/rhsm/candlepin_proxies/rhsm_index - katello/environments/autocomplete_search	Katello::KTEEnvironment
create_lifecycle_environments	katello/api/v2/environments/create	Katello::KTEEnvironment
edit_lifecycle_environments	katello/api/v2/environments/update	Katello::KTEEnvironment
destroy_lifecycle_environments	katello/api/v2/environments/destroy	Katello::KTEEnvironment
promote_or_remove_content_views_to_environments		Katello::KTEEnvironment

Permission Name	Actions	Resource Type
view_locations	- locations/index - locations/show - locations/auto_complete_search - api/v2/locations/index - api/v2/locations/show	
create_locations	- locations/new - locations/create - locations/clone_taxonomy - locations/step2 - locations/nest - api/v2/locations/create	
edit_locations	- locations/edit - locations/update - locations/import_mismatches - locations/parent_taxonomy_selected - api/v2/locations/update	
destroy_locations	- locations/destroy - api/v2/locations/destroy	
assign_locations	- locations/assign_all_hosts - locations/assign_hosts - locations/assign_selected_hosts	

Permission Name	Actions	Resource Type
view_mail_notifications	- mail_notifications/index - mail_notifications/auto_complete_search - mail_notifications/show - api/v2/mail_notifications/index - api/v2/mail_notifications/show	
view_media	- media/index - media/show - media/auto_complete_search - api/v2/media/index - api/v2/media/show	
create_media	- media/new - media/create - api/v2/media/create	
edit_media	- media/edit - media/update - api/v2/media/update	
destroy_media	- media/destroy - api/v2/media/destroy	

Permission Name	Actions	Resource Type
view_models	- models/index - models/show - models/auto_complete_search - api/v2/models/index - api/v2/models/show	
create_models	- models/new - models/create - api/v2/models/create	
edit_models	- models/edit - models/update - api/v2/models/update	
destroy_models	- models/destroy - api/v2/models/destroy	

Permission Name	Actions	Resource Type
view_operatingsystems	- operatingsystems/index - operatingsystems/show - operatingsystems/bootfiles - operatingsystems/auto_complete_search - api/v2/operatingsystems/index - api/v2/operatingsystems/show - api/v2/operatingsystems/bootfiles - api/v2/os_default_templates/index - api/v2/os_default_templates/show	
create_operatingsystems	- operatingsystems/new - operatingsystems/create - api/v2/operatingsystems/create - api/v2/os_default_templates/create	

Permission Name	Actions	Resource Type
edit_operatingsystems	● <code>operatingsystems/edit</code> ● <code>operatingsystems/update</code> ● <code>api/v2/operatingsystems/update</code> ● <code>api/v2/parameters/create</code> ● <code>api/v2/parameters/update</code> ● <code>api/v2/parameters/destroy</code> ● <code>api/v2/parameters/reset</code> ● <code>api/v2/os_default_templates/create</code> ● <code>api/v2/os_default_templates/update</code> ● <code>api/v2/os_default_templates/destroy</code>	
destroy_operatingsystems	● <code>operatingsystems/destroy</code> ● <code>api/v2/operatingsystems/destroy</code> ● <code>api/v2/os_default_templates/create</code>	

Permission Name	Actions	Resource Type
view_organizations	• organizations/index • organizations/show • organizations/auto_complete_search • api/v2/organizations/index • api/v2/organizations/show • katello/api/v2/organizations/index • katello/api/v2/organizations/show • katello/api/v2/organizations/redhat_provider • katello/api/v2/organizations/download_debug_certificate • katello/api/v2/tasks/index	
create_organizations	• organizations/new • organizations/create • organizations/clone_taxonomy • organizations/step2 • organizations/nest • api/v2/organizations/create • katello/api/v2/organizations/create	

Permission Name	Actions	Resource Type
edit_organizations	- organizations/edit - organizations/update - organizations/import_matches - organizations/parent_taxonomy_selected - api/v2/organizations/update - katello/api/v2/organizations/update - katello/api/v2/organizations/autoattach_subscriptions	
destroy_organizations	- organizations/destroy - api/v2/organizations/destroy - katello/api/v2/organizations/destroy	
assign_organizations	- organizations/assign_all_hosts - organizations/assign_hosts - organizations/assign_selected_hosts	
view_ptables	- ptables/index - ptables/show - ptables/auto_complete_search - ptables/revision - ptables/preview - api/v2/ptables/show - api/v2/ptables/revision	

Permission Name	Actions	Resource Type
create_ptables	● ptables/new ● ptables/create ● ptables/clone_template ● api/v2/ptables/create ● api/v2/ptables/clone	
edit_ptables	● ptables/edit ● ptables/update ● api/v2/ptables/update	
destroy_ptables	● ptables/destroy ● api/v2/ptables/destroy	
lock_ptables	● ptables/lock ● ptables/unlock ● api/v2/ptables/lock ● api/v2/ptables/unlock	
view_plugins	● plugins/index ● api/v2/plugins/index	
view_products	● katello/products/auto_complete ● katello/products/auto_complete_search ● katello/api/v2/products/index ● katello/api/v2/products/show ● katello/api/v2/repositories/index ● katello/api/v2/repositories/show	Katello::Product

Permission Name	Actions	Resource Type
	- katello/api/v2/packages/index - katello/api/v2/packages/show - katello/api/v2/distributions/index - katello/api/v2/distributions/show - katello/api/v2/package_groups/index - katello/api/v2/package_groups/show - katello/api/v2/errata/index - katello/api/v2/errata/show - katello/api/v2/puppet_modules/index - katello/api/v2/puppet_modules/show - katello/errata/short_details - katello/errata/auto_complete - katello/packages/details - katello/packages/auto_complete - katello/puppet_modules/show - katello/repositories/auto_complete_library - katello/repositories/repository_types - katello/content_search/index - katello/content_search/products - katello/content_search/repos - katello/content_search/packages - katello/content_search/errata	

Permission Name	Actions	Resource Type
	- katello/content_search/puppet_modules - katello/content_search/packages_items - katello/content_search/errata_items - katello/content_search/puppet_modules_items - katello/content_search/r epo_packages - katello/content_search/r epo_errata - katello/content_search/r epo_puppet_modules - katello/content_search/r epo_compare_errata - katello/content_search/r epo_compare_packages - katello/content_search/r epo_compare_puppet_modules	
create_products	- katello/api/v2/products/create - katello/api/v2/repositories/create	Katello::Product

Permission Name	Actions	Resource Type
edit_products	- katello/api/v2/products/update - katello/api/v2/repositories/update - katello/api/v2/repositories/remove_content - katello/api/v2/repositories/import_uploads - katello/api/v2/repositories/upload_content - katello/api/v2/products_bulk_actions/update_sync_plans - katello/api/v2/content_uploads/create - katello/api/v2/content_uploads/update - katello/api/v2/content_uploads/destroy - katello/api/v2/organizations/repo_discover - katello/api/v2/organizations/cancel_repo_discover	Katello::Product
destroy_products	- katello/api/v2/products/destroy - katello/api/v2/repositories/destroy - katello/api/v2/products_bulk_actions/destroy_products - katello/api/v2/repositories_bulk_actions/destroy_repositories	Katello::Product

Permission Name	Actions	Resource Type
sync_products	- katello/api/v2/products/sync - katello/api/v2/repositories/sync - katello/api/v2/products_bulk_actions/sync_products - katello/api/v2/repositories_bulk_actions/sync_repositories - katello/api/v2/sync/index - katello/api/v2/sync_plans/sync - katello/sync_management/index - katello/sync_management/sync_status - katello/sync_management/product_status - katello/sync_management/sync - katello/sync_management/destroy	Katello::Product
export_products	katello/api/v2/repositories/export	Katello::Product

Permission Name	Actions	Resource Type
view_provisioning_templates	- provisioning_templates/index - provisioning_templates/show - provisioning_templates/revision - provisioning_templates/auto_complete_search - provisioning_templates/preview - api/v2/provisioning_templates/index - api/v2/provisioning_templates/show - api/v2/provisioning_templates/revision - api/v2/template_combinations/index - api/v2/template_combinations/show - api/v2/template_kinds/index	
create_provisioning_templates	- provisioning_templates/new - provisioning_templates/create - provisioning_templates/clone_template - api/v2/provisioning_templates/create - api/v2/provisioning_templates/clone - api/v2/template_combinations/create	

Permission Name	Actions	Resource Type
edit_provisioning_templates	- provisioning_templates/edit - provisioning_templates/update - api/v2/provisioning_templates/update - api/v2/template_combinations/update	
destroy_provisioning_templates	- provisioning_templates/destroy - api/v2/provisioning_templates/destroy - api/v2/template_combinations/destroy	
deploy_provisioning_templates	- provisioning_templates/build_pxe_default - api/v2/provisioning_templates/build_pxe_default	
lock_provisioning_templates	- provisioning_templates/lock - provisioning_templates/unlock - api/v2/provisioning_templates/lock - api/v2/provisioning_templates/unlock	
user_logout	users/logout	
my_account	- users/edit - katello/api/v2/tasks/show	

Permission Name	Actions	Resource Type
api_status	• <code>api/v2/home/status/</code>	
view_puppetclasses	• <code>puppetclasses/index</code> • <code>puppetclasses/show</code> • <code>puppetclasses/auto_complete_search</code> • <code>api/v2/puppetclasses/index</code> • <code>api/v2/puppetclasses/show</code> • <code>api/v2/smart_variables/index</code> • <code>api/v2/smart_variables/show</code> • <code>api/v2/smart_class_parameters/index</code> • <code>api/v2/smart_class_parameters/show</code>	
create_puppetclasses	• <code>puppetclasses/new</code> • <code>puppetclasses/create</code> • <code>api/v2/puppetclasses/create</code>	

Permission Name	Actions	Resource Type
edit_puppetclasses	• puppetclasses/edit • puppetclasses/update • puppetclasses/override • api/v2/puppetclasses/update • api/v2/smart_variables/create • api/v2/smart_variables/update • api/v2/smart_variables/destroy • api/v2/smart_class_parameters/create • api/v2/smart_class_parameters/update • api/v2/smart_class_parameters/destroy	
destroy_puppetclasses	• puppetclasses/destroy • api/v2/puppetclasses/destroy	
import_puppetclasses	• puppetclasses/import_environments • puppetclasses/obsolete_and_new • api/v2/environments/import_puppetclasses • api/v2/smart_proxies/import_puppetclasses	

Permission Name	Actions	Resource Type
view_realms	● realms/index ● realms/show ● realms/auto_complete_search ● api/v2/realms/index ● api/v2/realms/show	
create_realms	● realms/new ● realms/create ● api/v2/realms/create	
edit_realms	● realms/edit ● realms/update ● api/v2/realms/update	
destroy_realms	● realms/destroy ● api/v2/realms/destroy	
view_search	● redhat_access/search/index	
view_cases	● redhat_access/cases/index ● redhat_access/cases/create	
attachments	● redhat_access/attachments/index ● redhat_access/attachments/create	

Permission Name	Actions	Resource Type
configuration	● redhat_access/configuration/index	
app_root	● redhat_access/redhat_access/index	
view_log_viewer	● redhat_access/logviewer/index	
logs	● redhat_access/logs/index	
rh_telemetry_api	● redhat_access/api/telemetry_api/proxy ● redhat_access/api/telemetry_api/connection_status	
rh_telemetry_view	● redhat_access/analytics_dashboard/index	
rh_telemetry_configurations	● redhat_access/telemetry_configurations/show ● redhat_access/telemetry_configurations/update	
view_roles	● roles/index ● roles/auto_complete_search ● api/v2/roles/index ● api/v2/roles/show	

Permission Name	Actions	Resource Type
create_roles	- roles/new - roles/create - roles/clone - api/v2/roles/create	
edit_roles	- roles/edit - roles/update - api/v2/roles/update	
destroy_roles	- roles/destroy - api/v2/roles/destroy	
access_settings	home/settings	

Permission Name	Actions	Resource Type
view_smart_proxies	● smart_proxies/index ● smart_proxies/ping ● smart_proxies/auto_complete_search ● smart_proxies/version ● smart_proxies/show ● smart_proxies/plugin_version ● smart_proxies/tftp_server ● smart_proxies/puppet_environments ● smart_proxies/puppet_dashboard ● smart_proxies/log-pane ● smart_proxies/failed_modules ● smart_proxies/errors_card ● smart_proxies/modules_card ● api/v2/smart_proxies/index ● api/v2/smart_proxies/show ● api/v2/smart_proxies/version ● api/v2/smart_proxies/log	
create_smart_proxies	● smart_proxies/new ● smart_proxies/create ● api/v2/smart_proxies/create	

Permission Name	Actions	Resource Type
edit_smart_proxies	- smart_proxies/edit - smart_proxies/update - smart_proxies/refresh - smart_proxies/expire_logs - api/v2/smart_proxies/update - api/v2/smart_proxies/refresh	
destroy_smart_proxies	- smart_proxies/destroy - api/v2/smart_proxies/destroy	
view_smart_proxies_autosign	- autosign/index - autosign/show - autosign/counts - api/v2/autosign/index	
create_smart_proxies_autosign	- autosign/new - autosign/create	
destroy_smart_proxies_autosign	autosign/destroy	
view_smart_proxies_puppetca	- puppetca/index - puppetca/counts - puppetca/expiry	
edit_smart_proxies_puppetca	puppetca/update	
destroy_smart_proxies_puppetca	puppetca/destroy	

Permission Name	Actions	Resource Type
view_subnets	• subnets/index • subnets/show • subnets/auto_complete_search • api/v2/subnets/index • api/v2/subnets/show	
create_subnets	• subnets/new • subnets/create • api/v2/subnets/create	
edit_subnets	• subnets/edit • subnets/update • api/v2/subnets/update	
destroy_subnets	• subnets/destroy • api/v2/subnets/destroy	
import_subnets	• subnets/import • subnets/create_multiple	

Permission Name	Actions	Resource Type
view_subscriptions	- katello/api/v2/subscriptions/index - katello/api/v2/subscriptions/show - katello/api/v2/subscriptions/available - katello/api/v2/subscriptions/manifest_history - katello/api/v2/subscriptions/auto_complete_search - katello/api/v2/repository_sets/index - katello/api/v2/repository_sets/show - katello/api/v2/repository_sets/available_repositories	Organization
attach_subscriptions	katello/api/v2/subscriptions/create	Organization
unattach_subscriptions	katello/api/v2/subscriptions/destroy	Organization

Permission Name	Actions	Resource Type
import_manifest	- katello/products/available_repositories - katello/products/toggle_repository - katello/providers/redhat_provider - katello/providers/redhat_provider_tab - katello/api/v2/subscriptions/upload - katello/api/v2/subscriptions/refresh_manifest - katello/api/v2/repository_sets/enable - katello/api/v2/repository_sets/disable	Organization
delete_manifest	katello/api/v2/subscriptions/delete_manifest	Organization
view_sync_plans	- katello/sync_plans/all - katello/sync_plans/index - katello/sync_plans/auto_complete_search - katello/api/v2/sync_plans/index - katello/api/v2/sync_plans/show - katello/api/v2/sync_plans/add_products - katello/api/v2/sync_plans/remove_products - katello/api/v2/sync_plans/available_products - katello/api/v2/products/index	Katello::SyncPlan

Permission Name	Actions	Resource Type
create_sync_plans	katello/api/v2/sync_plans/create	Katello::SyncPlan
edit_sync_plans	katello/api/v2/sync_plans/update	Katello::SyncPlan
destroy_sync_plans	katello/api/v2/sync_plans/destroy	Katello::SyncPlan
	my_organizations	katello/api/rhsm/candlepin_proxies/list_owners
	view_usergroups	- usergroups/index - usergroups/show - usergroups/auto_complete_search - api/v2/usergroups/index - api/v2/usergroups/show
	create_usergroups	- usergroups/new - usergroups/create - api/v2/usergroups/create
	edit_usergroups	- usergroups/edit - usergroups/update - api/v2/usergroups/update
	destroy_usergroups	- usergroups/destroy - api/v2/usergroups/destroy

Permission Name	Actions	Resource Type
	view_users	• users/index • users/show • users/auto_complete_search • api/v2/users/index • api/v2/users/show
	create_users	• users/new • users/create • users/auth_source_selected • api/v2/users/create
	edit_users	• users/edit • users/update • users/auth_source_selected • users/test_mail • api/v2/users/update
	destroy_users	• users/destroy • api/v2/users/destroy