



Red Hat CloudForms 4.1

Red Hat CloudForms REST API

Using the Red Hat CloudForms REST API

Red Hat CloudForms 4.1 Red Hat CloudForms REST API

Using the Red Hat CloudForms REST API

Red Hat CloudForms Documentation Team
cloudforms-docs@redhat.com

Legal Notice

Copyright © 2018 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, 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 Software Collections 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

This guide provides web services available to integrate Red Hat CloudForms with external applications. It details the specification of the Red Hat CloudForms RESTful API, which is implemented as standard REST HTTP requests and responses of content type JSON. If you have a suggestion for improving this guide or have found an error, please submit a Bugzilla report at <http://bugzilla.redhat.com> against Red Hat CloudForms Management Engine for the Documentation component. Please provide specific details, such as the section number, guide name, and CloudForms version so we can easily locate the content.

Table of Contents

| | |
|---|-----------|
| CHAPTER 1. SPECIFICATION | 7 |
| 1.1. HTTP BASICS | 7 |
| 1.1.1. REST API Entry Point | 7 |
| 1.1.2. Supported Content Types | 7 |
| 1.1.3. URL Paths | 8 |
| 1.1.4. Methods and related URLs | 8 |
| 1.1.5. Updating Resources | 9 |
| 1.1.6. Modifying Resource Attributes | 9 |
| 1.1.7. Return Codes | 9 |
| 1.1.8. CRUD Examples | 10 |
| 1.2. AUTHENTICATION | 11 |
| 1.2.1. Using Basic Authentication | 12 |
| 1.2.2. Using Authentication Tokens | 12 |
| 1.3. JSON SPECIFICATION | 13 |
| 1.3.1. Basic types | 13 |
| 1.3.2. Common Attributes and Actions | 13 |
| 1.3.3. Collections | 14 |
| 1.3.4. Action Specification | 15 |
| 1.3.5. Forms | 17 |
| 1.4. QUERY SPECIFICATION | 18 |
| 1.4.1. Control Attributes | 18 |
| 1.4.2. Filtering | 19 |
| 1.4.3. Expanding Collections | 19 |
| CHAPTER 2. REFERENCE GUIDE | 21 |
| 2.1. AUTHENTICATION | 21 |
| 2.2. HTTP HEADERS | 21 |
| 2.3. LISTING AND QUERYING COLLECTIONS AND SUB-COLLECTIONS | 21 |
| 2.4. COLLECTION QUERIES | 22 |
| 2.5. SUB-COLLECTION QUERIES | 23 |
| 2.6. AVAILABLE ACTIONS | 24 |
| 2.7. PROVISIONING REQUEST ATTRIBUTES | 27 |
| 2.7.1. Provisioning Request Attribute Groups | 28 |
| 2.7.2. Service Catalog Attributes | 29 |
| 2.7.3. Hardware Attributes | 29 |
| 2.7.4. Network Attributes | 30 |
| 2.7.5. Custom Attributes | 30 |
| 2.7.6. Schedule Attributes | 32 |
| 2.7.7. Requester Attributes | 33 |
| 2.7.8. Environment Attributes | 34 |
| CHAPTER 3. EXAMPLES | 36 |
| 3.1. GENERAL QUERIES | 36 |
| 3.1.1. Queries | 36 |
| 3.1.2. Paging Queries | 37 |
| 3.1.2.1. Request: | 37 |
| 3.1.2.2. Response: | 37 |
| 3.1.2.3. Request: | 37 |
| 3.1.2.4. Response: | 37 |
| 3.1.2.5. Request: | 38 |
| 3.1.2.6. Response: | 38 |
| 3.1.2.7. Request: | 38 |

| | |
|---|----|
| 3.1.2.8. Response: | 39 |
| 3.1.3. Querying a Delete Task | 39 |
| 3.1.3.1. Request: | 39 |
| 3.1.3.2. Response: | 39 |
| 3.2. SERVICE CATALOGS | 40 |
| 3.2.1. Adding a Sample Service Catalog | 40 |
| 3.2.1.1. Request: | 40 |
| 3.2.1.2. Response: | 40 |
| 3.2.2. Adding Multiple Service Catalogs | 40 |
| 3.2.2.1. Request: | 40 |
| 3.2.2.2. Response: | 41 |
| 3.2.3. Assigning Service Templates to Service Catalogs | 41 |
| 3.2.3.1. Request: | 41 |
| 3.2.3.2. Response: | 42 |
| 3.2.4. Editing a Service Catalog | 42 |
| 3.2.4.1. Request: | 42 |
| 3.2.4.2. Response: | 42 |
| 3.2.5. Editing Multiple Service Catalogs | 43 |
| 3.2.5.1. Request: | 43 |
| 3.2.5.2. Response: | 43 |
| 3.2.6. Ordering a Single Service from a Service Catalog | 43 |
| 3.2.6.1. Request: | 43 |
| 3.2.6.2. Response: | 44 |
| 3.2.7. Ordering Multiple Services from a Service Catalog | 44 |
| 3.2.7.1. Request: | 44 |
| 3.2.7.2. Response: | 45 |
| 3.2.8. Unassigning Service Templates from a Service Catalog | 46 |
| 3.2.8.1. Request: | 46 |
| 3.2.8.2. Response: | 47 |
| 3.2.9. Deleting Multiple Service Catalogs | 47 |
| 3.2.9.1. Request: | 47 |
| 3.2.9.2. Response: | 47 |
| 3.3. TAGS | 48 |
| 3.3.1. Assigning Tags to a Service | 48 |
| 3.3.1.1. Request: | 48 |
| 3.3.1.2. Response: | 48 |
| 3.3.2. Assigning Tags by Name to a Service | 49 |
| 3.3.2.1. Request: | 49 |
| 3.3.2.2. Response: | 49 |
| 3.3.3. Assigning Tags by Reference to a Service | 49 |
| 3.3.3.1. Request: | 49 |
| 3.3.3.2. Response: | 50 |
| 3.3.4. Assigning Tags to a Service Template | 50 |
| 3.3.4.1. Request: | 50 |
| 3.3.4.2. Response: | 50 |
| 3.3.5. Assigning Tags to a Virtual Machine | 51 |
| 3.3.5.1. Request: | 51 |
| 3.3.5.2. Response: | 51 |
| 3.3.6. Assigning Tags by Name to a Virtual Machine | 51 |
| 3.3.6.1. Request: | 51 |
| 3.3.6.2. Response: | 52 |
| 3.3.7. Assigning Tags by Reference to a Virtual Machine | 52 |
| 3.3.7.1. Request: | 52 |

| | |
|---|----|
| 3.3.7.2. Response: | 52 |
| 3.3.8. Unassigning Tags from a Service | 53 |
| 3.3.8.1. Request: | 53 |
| 3.3.8.2. Response: | 53 |
| 3.3.9. Unassigning a Tag by Name from a Service | 53 |
| 3.3.9.1. Request: | 53 |
| 3.3.9.2. Response: | 54 |
| 3.3.10. Unassigning a Tag by Reference from a Service | 54 |
| 3.3.10.1. Request: | 54 |
| 3.3.10.2. Response: | 54 |
| 3.3.11. Unassigning Tags from a Service Template | 54 |
| 3.3.11.1. Request: | 54 |
| 3.3.11.2. Response: | 55 |
| 3.4. AUTOMATION REQUESTS | 55 |
| 3.4.1. Triggering a Single Automation Request | 55 |
| 3.4.1.1. Request: | 55 |
| 3.4.1.2. Request: | 56 |
| 3.4.1.3. Response: | 56 |
| 3.4.2. Triggering Multiple Automation Requests | 57 |
| 3.4.2.1. Request: | 57 |
| 3.4.2.2. Response: | 59 |
| 3.5. PROVISIONING REQUESTS | 61 |
| 3.5.1. Triggering a Single Provision Request | 61 |
| 3.5.1.1. Request: | 61 |
| 3.5.1.2. Request: | 62 |
| 3.5.1.3. Response: | 62 |
| 3.5.2. Triggering Multiple Provision Requests | 65 |
| 3.5.2.1. Request: | 65 |
| 3.5.2.2. Response: | 67 |
| 3.5.3. Monitoring Request | 75 |
| 3.5.3.1. Response: | 75 |
| 3.6. PROVIDERS | 76 |
| 3.6.1. Creating a Provider | 76 |
| 3.6.1.1. Request: | 76 |
| 3.6.1.2. Response: | 76 |
| 3.6.2. Creating a Provider with Compound Credentials | 77 |
| 3.6.2.1. Request: | 77 |
| 3.6.2.2. Response: | 77 |
| 3.6.3. Refreshing a Provider | 78 |
| 3.6.3.1. Request: | 78 |
| 3.6.3.2. Response: | 78 |
| 3.6.4. Updating a Provider | 78 |
| 3.6.4.1. Request: | 78 |
| 3.6.4.2. Response: | 78 |
| 3.6.5. Deleting a Provider | 79 |
| 3.6.5.1. Deleting a Single Provider | 79 |
| 3.6.6. Request: | 79 |
| 3.6.7. Response: | 79 |
| 3.6.8. Request: | 79 |
| 3.6.9. Response: | 79 |
| 3.6.9.1. Deleting Multiple Providers | 80 |
| 3.6.10. Request: | 80 |
| 3.6.11. Response: | 80 |

| | |
|--|----|
| 3.7. SERVICES | 80 |
| 3.7.1. Editing a Service | 80 |
| 3.7.1.1. Request: | 80 |
| 3.7.1.2. Response: | 81 |
| 3.7.2. Editing Multiple Services | 81 |
| 3.7.2.1. Request: | 81 |
| 3.7.2.2. Response: | 81 |
| 3.7.3. Editing a Resource with the PATCH Method | 82 |
| 3.7.3.1. Request: | 82 |
| 3.7.3.2. Response: | 82 |
| 3.7.4. Editing a Service with the PUT Method | 82 |
| 3.7.4.1. Request: | 82 |
| 3.7.4.2. Response: | 83 |
| 3.7.5. Retiring a Service Immediately | 83 |
| 3.7.5.1. Request: | 83 |
| 3.7.5.2. Response: | 83 |
| 3.7.5.3. Response: | 84 |
| 3.7.6. Retiring a Service at a Future Time | 84 |
| 3.7.6.1. Request: | 84 |
| 3.7.6.2. Response: | 84 |
| 3.7.7. Retiring Multiple Services | 84 |
| 3.7.7.1. Request: | 84 |
| 3.7.7.2. Response: | 85 |
| 3.7.8. Deleting Services | 86 |
| 3.7.8.1. Request: | 86 |
| 3.7.8.2. Response: | 86 |
| 3.8. VIRTUAL MACHINES | 86 |
| 3.8.1. Scanning a Virtual Machine | 86 |
| 3.8.1.1. Request: | 86 |
| 3.8.1.2. Response: | 87 |
| 3.8.1.3. Request: | 87 |
| 3.8.1.4. Response: | 87 |
| 3.8.2. Setting the Virtual Machine Owner | 87 |
| 3.8.2.1. Request: | 87 |
| 3.8.2.2. Response: | 87 |
| 3.8.3. Adding an Event to a Virtual Machine | 88 |
| 3.8.3.1. Request: | 88 |
| 3.8.3.2. Response: | 88 |
| 3.8.4. Adding a Lifecycle Event to a Virtual Machine | 88 |
| 3.8.4.1. Request: | 88 |
| 3.8.4.2. Response: | 88 |
| 3.8.5. Starting a Virtual Machine | 89 |
| 3.8.5.1. Request: | 89 |
| 3.8.5.2. Response: | 89 |
| 3.8.6. Querying Task Progress | 89 |
| 3.8.6.1. Request: | 89 |
| 3.8.6.2. Response: | 89 |
| 3.8.7. Stopping a Virtual Machine | 89 |
| 3.8.7.1. Request: | 89 |
| 3.8.7.2. Response: | 90 |
| 3.8.8. Suspending a Virtual Machine | 90 |
| 3.8.8.1. Request: | 90 |
| 3.8.8.2. Response: | 90 |

| | |
|--|----|
| 3.8.9. Deleting Virtual Machines | 90 |
| 3.8.9.1. Deleting a Single Virtual Machine | 90 |
| 3.8.10. Request: | 90 |
| 3.8.10.1. Deleting Multiple Virtual Machines | 90 |
| 3.8.11. Request: | 91 |
| 3.8.12. Response: | 91 |
| 3.8.13. Request: | 91 |
| 3.8.14. Response: | 91 |
| 3.9. SERVICE TEMPLATES | 92 |
| 3.9.1. Editing a Service Template | 92 |
| 3.9.1.1. Request: | 92 |
| 3.9.1.2. Response: | 92 |
| 3.9.2. Editing Multiple Service Templates | 92 |
| 3.9.2.1. Request: | 92 |
| 3.9.2.2. Response: | 93 |
| 3.9.3. Deleting Multiple Service Templates | 93 |
| 3.9.3.1. Request: | 93 |
| 3.9.3.2. Response: | 94 |

CHAPTER 1. SPECIFICATION

1.1. HTTP BASICS

1.1.1. REST API Entry Point

The REST API is available via the `/api` URL prefix. It is accessed on the Red Hat CloudForms server as follows:

```
https://<host_fqdn>/api
```

Response

```
{
  "name" : "API",
  "description" : "REST API",
  "version" : "2.0.0",
  "versions" : [
    {
      "name" : "2.0.0",
      "href" : "https://hostname/api/v2.0.0"
    },
  ],
  "collections" : [
    {
      "name" : "automation_requests",
      "href" : "https://hostname/api/automation_requests",
      "description" : "Automation Requests"
    },
    {
      "name" : "availability_zones",
      "href" : "https://hostname/api/availability_zones",
      "description" : "Availability Zones"
    },
    ...
  ]
}
```

- **version** is the current API version, accessible via either of the following:
 - `/api/`
 - `/api/v2.0.0/`
- **versions** lists all the earlier API versions that are still exposed via their respective entry points:
 - `/api/vVersion/`

1.1.2. Supported Content Types

Requests:

Accept: application/json

Responses:

Content-Type: application/json

1.1.3. URL Paths

The recommended convention for URLs is to use alternative collection or resource path segments, relative to the API entry point as described in the following example:

| URL | Description |
|-------------------------------------|--|
| /api | The REST API Entrypoint |
| /api/vVersion | The REST Entrypoint for a specific version of the REST API |
| /api/:collection | A top-level collection |
| /api/:collection/:id | A specific resource of that collection |
| /api/:collection/:id/:subcollection | Sub-collection under the specific resource |

1.1.4. Methods and related URLs

The basic HTTP Methods used for the API are GET, POST, PUT, PATCH and DELETE.

| URL | Semantic |
|-----------------------------|---|
| GET /api/:collection | Return all resources of the collection |
| GET /api/:collection/:id | Return the specific resource |
| POST /api/:collection | Create a resource in the collection |
| POST /api/:collection/:id | Perform an Action on a resource in the collection |
| PUT /api/:collection/:id | Update a specific resource |
| PATCH /api/:collection/:id | Update a specific resource |
| DELETE /api/:collection/:id | Delete a specific resource |

:collection represents specific entities such as services, hosts, and virtual machines.

1.1.5. Updating Resources

Use the following methods to update attributes in a resource:

- Update a resource via the **PUT** HTTP Method
- Update a resource via a **POST** Method with an **edit** action.
- Update a resource via the **PATCH** HTTP Method

While **PUT** is the common method, the **PATCH** mechanism gives better control on which attribute to edit or add, and enables removal, which is not available with the other two methods.

1.1.6. Modifying Resource Attributes

PUT /api/vms/42

```
{
  "name" : "A new VM name",
  "description" : "A Description for the new VM"
}
```

POST /api/vms/42

```
{
  "action" : "edit",
  "resource" : {
    "name" : "A new VM name",
    "description" : "A Description for the new VM"
  }
}
```

PATCH /api/vms/42

```
[
  { "action": "edit", "path": "name", "value": "A new VM name" },
  { "action": "add", "path": "description", "value": "A Description for
the new VM" },
  { "action": "remove", "path": "policies/3/description" }
]
```

In the **PATCH** implementation, path either references local attributes or attributes from a related resource in a subcollection.

1.1.7. Return Codes

Success

- **200 OK** - The request has succeeded without errors, this code should be returned for example when retrieving a collection or a single resource.
- **201 Created** - The request has been fulfilled and resulted in a **new resource being created**. The resource is available before this status code is returned. The response includes the HTTP body of the newly created resource.

- **202 Accepted** - The request has been accepted for processing, but the processing has not been completed. Like, resource is not fully available yet. This status code is usually returned when the resource creation happens asynchronously. In this case the HTTP response includes a pointer to **monitor** or a **job** where the client can query to get the current status of the request and the estimate on when the request will be actually fulfilled.
- **204 No Content** - The server has fulfilled the request but does not need to return an entity-body, and might want to return updated meta information. This HTTP response is commonly used for the **DELETE** requests, as the resource that was deleted does not exist anymore.

Client Errors

- **400 Bad Request** - The request could not be understood by the server due to malformed syntax. The client **SHOULD NOT** repeat the request without modifications. In REST API this status code should be returned to client when the client use the wrong combination of attributes, like expanding the non-existing collection, or using the pagination parameter incorrectly. Another use-case could be creating or performing actions on the resource, when the wrong JSON serialization of the resource or action is used.
- **401 Unauthorized** - The request requires user authentication. The response **MUST** include a **Authenticate** header field containing a challenge applicable to the requested resource. If the request include **Authenticate** header, then this HTTP status code might indicate that the current user is **not authorized** to perform given action or to access given resource.
- **403 Forbidden** - The server understood the request, but is refusing to fulfill it. Authorization will not help in this case. This HTTP status code might indicate that the action performed is not supported for this resource or collection.
- **404 Not Found** - In this case, the server has not found anything that matches with the URL.
- **415 Unsupported Media Type** - The server is refusing to service the request because the entity of the request is in a format not supported by the requested resource for the requested method. This error must be returned, when the client is explicitly asking for format other than JSON (application/json).

Server Errors

- **500 Internal Server Error** - The server encountered an unexpected condition which prevented it from fulfilling the request. This error code must be used when an exception is raised in the application and the exception has nothing to do with the client request.

1.1.8. CRUD Examples

The following examples show the basic CRUD operations (Create, Read, Update, Delete) using the REST API.

The commands below use basic authentication via the `--user admin:smartvm` credentials argument. For multiple API calls, it is recommended to access the Red Hat CloudForms REST API via token-based authentication. See [Authentication](#) for details.

Show a Collection of Resources

Get a collection of services: **GET /api/services**

```
curl --user admin:smartvm
-i -X GET -H "Accept: application/json"
https://hostname/api/services
```

Return a Single Resource

Return a single service: **GET /api/services/:id**

```
curl --user admin:smartvm
-i -X GET -H "Accept: application/json"
https://hostname/api/services/1
```

Create a Resource

Create a new provider: **POST /api/providers**

```
curl --user admin:smartvm
-i -X POST -H "Accept: application/json"
-d '{
  "type"      : "ManageIQ::Providers::Redhat::InfraManager",
  "name"      : "RHEVM Provider",
  "hostname"  : "rhevm.local.com",
  "ipaddress" : "192.168.5.1",
  "credentials" : {
    "userid"   : "admin",
    "password" : "12345"
  }
}'
https://hostname/api/providers
```

Update a Resource

Update the name of a service: **PUT /api/services/:id**

```
curl --user admin:smartvm
-i -X PUT -H "Accept: application/json"
-d '{"name" : "updated service name" }'
https://hostname/api/services/1
```

Delete a Resource

Delete a service: **DELETE /api/services/:id**

```
curl --user admin:smartvm
-i -X DELETE -H "Accept: application/json"
https://hostname/api/services/1
```

1.2. AUTHENTICATION

There are two methods of authentication for the Red Hat CloudForms REST API:

- **Basic Authentication:** The user and password credentials are passed in with each HTTP request.

- **Token based Authentication:** The client requests a token for the username/password credentials specified. Then the token is used in lieu of the username/password for each subsequent API call.

1.2.1. Using Basic Authentication

The following example demonstrates how to use basic authentication:

```
$ curl --user username:password
-i -X GET -H "Accept: application/json"
https://hostname/api/services/1013
```

Red Hat recommends token-based authentication for multiple REST API calls.

1.2.2. Using Authentication Tokens

Authentication Tokens:

- Are associated with the user credential.
- Provide the necessary identification for RBAC in subsequent REST calls.
- Expire after a certain amount of time (10 minutes by default).

Request

```
$ curl --user username:password
-i X GET -H "Accept: application/json"
https://hostname/api/auth
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "auth_token" : "af0245-238722-4d23db",
  "expires_on" : "2013-12-07T18:20:07Z"
}
```

Request using Token based authentication

```
$ curl -i -X GET -H "Accept: application/json"
-H "X-Auth-Token: af0245-238722-4d23db"
https://hostname/api/services/1013
```

Failed response due to invalid token

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="Application"
...
```


When a request fails due to an invalid token, the client must re-authenticate with the user credentials to obtain a new Authentication Token.

1.3. JSON SPECIFICATION

The API uses JSON throughout; the Content-Type for all requests and responses is `application/json`.

As is general practice with REST, clients should not make assumptions about the server's URL space. Clients are expected to discover all URL's by navigating the API. To keep this document readable, we still mention specific URL's, generally in the form of an absolute path. Clients should not use these, or assume that the actual URL structure follows these examples, and instead use discovered URL's. Any client should start its discovery with the API entry point, here denoted with `/api`.

1.3.1. Basic types

The following are basic data types and type combinators that are used throughout:

| Name | Explanation | Example serialization |
|-------------------|---|---|
| Integer | Integer value | <code>{ "id" : 10 }</code> |
| String | JSON string | <code>{ "state" : "running" }</code> |
| URL | Absolute URL | <code>{ "href" : "http://SERVER/vms/1/start" }</code> |
| Timestamp | Timestamp in ISO8601 format | <code>{ "created" : "2013-12- 05T08:15:30Z" }</code> |
| Array[T] | Array where each entry has type T | <code>{ "vms" : [{ "id" : 1 }, { "id" : 2 }] }</code> |
| Ref[T] | A reference to a T, used to model relations, the T is a valid Resource identifier | <code>{ "vm" : { "href" : URL } }</code> |
| Collection | Array[T] where T represents a Ref[T], this might allow actions to be executed on all members as a single unit | <code>{ "vms" : { "count" : 2, "resources" : [{ "href" : URL }, { "href" : URL }], "actions" : [] } }</code> |
| Struct | A structure with sub-attributes | <code>"power_state": { "state" : "ON", "last_boot_time" : "2013-05- 29T15:28Z", "state_change_time" : "2013-05- 29T15:28Z" }</code> |

1.3.2. Common Attributes and Actions

The following describes attributes and actions that are shared by all resources and collections defined in this API.

Table 1.1. Attributes

| Attribute | Type | Description |
|-----------|-----------|---|
| id | Integer | An integer identifier for the referenced resource |
| href | Ref(self) | A unique self reference |
| name | String | A human name of the resource |

```
{
  "href" : "https://hostname/api/resources/1",
  "id" : 1,
  "name" : "first_resource"
}
```

Table 1.2. Actions

| Action | HTTP method | Description |
|--------|----------------|---------------------------------------|
| create | POST | Create new resource in the collection |
| edit | PUT/PATCH/POST | Edit attributes in resource |
| delete | DELETE | Delete resource |



NOTE

The availability of these common actions depends on the role and permissions that the current API user has for a particular resource.

1.3.3. Collections

Resources can be grouped into collections. Each collection is unordered, and is homogeneous so that it contains only one type of resource. Resources can also exist outside any collection; these resources are referred to as singleton resources. Collections are themselves resources as well.

Collections can exist globally, at the top level of an API, and can also be contained inside a single resource. The latter are referred to as sub-collections. Sub-collections are usually used to express a relationship where one resource is contained within another.

Collections are serialized in JSON in the following way:

```
{
  "name" : "String",
  "count": String,
}
```

```

    "subcount": String,
    "resources": [ ... ],
    "actions": [ ... ]
  }

```

- The **count** attribute in a collection always denotes the total number of items in the collection, not the number of items returned.
- The **subcount** attribute in a collection depicts the number of items returned.
- The **resources** attribute is an Array[T], where T might be a list of references to the T or, if expanded, a list of resources with all attributes.
- The **actions** attribute contains an Array of actions that can be performed against the collection resources.

1.3.4. Action Specification

The representation of each resource will only contain an action and its URL if the current user is presently allowed to perform that action against that resource. Actions will be contained in the **actions** attribute of a resource; that attribute contains an array of action definition, where each action definition has a **rel**, **method** and a **href** attribute.

- **name** attribute contains the action name
- **method** attribute states the HTTP method that must be used in a client HTTP request in order to perform the given action (eg. GET, POST, PUT, DELETE)
- **href** attribute contains the absolute URL that the HTTP request should be performed against
- **form** is an optional attribute that references a JSON document which describes the resource attributes that can be provided in the message body when performing this action. This description indicates which of those attributes are mandatory and which are optional.

Collection actions

The actions performed against a collection of resources, are in most cases batch operations against multiple resources. The action request must include an HTTP body with the action name and the list of resource representations that the action will be performed against.

The resource representation might include the resource attributes as they can change the way how the action is actually performed. In the example below, the first resource is started with **enable_ipmi** attribute, but the second resource omits this attribute which means the default value will be used.

Sample JSON request body for collection action:

POST /api/vms

```

{
  "action": "start",
  "resources" : [
    { "href" : "https://hostname/api/vms/1", "enable_ipmi" : "enabled",
      "initial_state" : "started" },
    { "href" : "https://hostname/api/vms/2" }
  ]
}

```

Actions in collection:

```

{
  "name" : "String",
  "count": String,
  "subcount": String,
  "resources": [ ... ],
  "actions": [
    {
      "name" : "shutdown",
      "method" : "post",
      "href" : "URL"
    },
    {
      "name" : "restart",
      "method" : "post",
      "href" : "URL"
    },
    {
      "name" : "poweron",
      "method" : "post",
      "href" : "URL"
    },
    {
      "name" : "poweroff",
      "method" : "post",
      "href" : "URL"
    },
    {
      "name" : "suspend",
      "method" : "post",
      "href" : "URL"
    },
    {
      "name" : "edit",
      "method" : "post",
      "form" : { "href" : "https://hostname/api/vms?form_for=add" },
      "href" : "URL"
    },
    {
      "name" : "destroy",
      "method" : "delete",
      "href" : "URL"
    }
  ]
}

```

Resource actions

An action performed against a given resource is always described in the body of the HTTP request. The HTTP body could contain a list of resource attributes that dictate how the state of the receiving resource is to be changed once the action is performed. At minimum the JSON document in the message body must contain the name of the action to be performed.

In cases where no attributes are required to perform an action the HTTP body will contain an empty JSON document, in which case default values will be assigned to the corresponding attributes.

Sample JSON request body for resource action:

POST /api/vms/123

```
{
  "action" : "start",
  "resource" : { "enable_ipmi" : "enabled" }
}
```

POST /api/vms/321

```
{
  "action" : "start",
  "resource" : {}
}
```

Actions in a resource:

```
{
  "href" : "Ref(self)",
  "id" : Integer,
  "name" : "resource human name",
  "actions" : [
    {
      "name" : "edit",
      "method" : "post",
      "form" : { "href" : "https://hostname/api/vms?form_for=edit" },
      "href" : "URL"
    }
  ]
}
```

1.3.5. Forms

Getting a Form

The URL to fetch a form is part of the `action` serialization. In a case when no form is referenced, the action does not require any attributes to be performed.

Resource including an action with a Form:

```
{
  "href" : "Ref(self)",
  "id" : Integer,
  "name" : "resource human name",
  "actions": [
    {
      "name" : "edit",
      "method" : "post",
      "form" : { "href" : "https://hostname/vms?form_for=edit" },
      "href" : "URL"
    }
  ]
}
```

```
GET /api/vms?form_for=edit HTTP/1.1
```

Example of a Form:

```
{
  "required" : [ "name", "host" ],
  "optional" : [ "description" ]
  "internal" : [ "power_state", "created_on" ]
}
```

The following describes the semantics of the attribute identifiers:

- **required** - These attributes must be specified for the action to be carried out.
- **optional** - These are optional attributes, which may be specified and processed by the action. These may be shown in a UI but not enforced.
- **internal** - It is not necessary to define these, but they are required for a UI form to show and extended a form with more attributes than the required and optional identifiers permit. This identifier shows what attributes are system managed and not modifiable by the REST client.

1.4. QUERY SPECIFICATION

This specification identifies the controls available when querying collections.

1.4.1. Control Attributes

The controls are specified in the GET URL as attribute value pairs as follows:

```
GET /api/resources?ctl1=val1&ctl2=val2
```

| Category | Attribute | Semantics |
|----------|--------------------------|---|
| Paging | | |
| | offset | 0-based offset of first item to return |
| | limit | number of items to return. If 0 is specified then the remaining items are returned |
| Scope | | |
| | filter[] | One or more filters to search on. See Filtering below. |
| | attributes=atr1,atr2,... | Which attributes in addition to id and href to return. If not specified or <i>all</i> (default is <i>attributes=all</i>), then all attributes are returned |

| Category | Attribute | Semantics |
|----------|------------------------------------|--|
| | expand=resources | To expand the resources returned in the collection and not just the href. See Expanding Collection below |
| Sorting | | |
| | sort_by=atr1,atr2,... | By which attribute(s) to sort the result by |
| | sort_order=ascending or descending | Order of the sort |

- The **count** attribute in a collection always denotes the total number of items in the collection, not the number of items returned.
- The **subcount** attribute in a collection denotes the number of items from the collection that were returned, for example as the result of a paged request.

1.4.2. Filtering

GET requests against collections support the following query parameters to enable filtering:

- **filter[]**: The SQL filter to use for querying the collection.

```
GET /api/resources?filter[]=name='myservice%25'
```

The query above requests resources that begin with the name **myservice**. String values must be contained in single or double quotes. Special characters within the quotes must be URL encoded. In the example above, the database wildcard character, %, is encoded as **%25**.

1.4.3. Expanding Collections

While in the JSON serialization example the description says that the resource might be a list of references to the resource, using the **expand** parameter returns a full JSON serialization of the resource instead:

```
GET /api/vms
```

```
{
  "name" : "vms"
  "count": 2,
  "subcount": 2,
  "resources": [
    { "href" : "https://hostname/api/vms/1" },
    { "href" : "https://hostname/api/vms/2" }
  ],
  "actions": []
}
```

GET /api/vms?expand=resources

```
{
  "name" : "vms"
  "count": 2,
  "subcount": 2,
  "resources": [
    {
      "href" : "https://hostname/api/vms/1",
      "id" : 1,
      "name" : "My First VM",
      ...
    },
    {
      "href" : "https://hostname/api/vms/2",
      "id" : 2,
      "name" : "My Second VM",
      ...
    }
  ],
  "actions": []
}
```


CHAPTER 2. REFERENCE GUIDE

2.1. AUTHENTICATION

| Type | Mechanism |
|-----------------------------|---|
| Basic Authentication | Basic HTTP Authorization with user and password |
| Token Based Authentication | |
| - Acquiring Token | /api/auth with Basic Authentication |
| - Authenticating with Token | X-Auth-Token Header |

2.2. HTTP HEADERS

| Header | Value |
|---------------|-------------------------------------|
| Authorization | Basic base64_encoded(user:password) |
| X-Auth-Token | Token provided by /api/auth |
| Accept | application/json |
| Content-Type | application/json |

2.3. LISTING AND QUERYING COLLECTIONS AND SUB-COLLECTIONS

| Feature | Path |
|-------------------------------|---|
| Listing Available Collections | /api |
| Listing Collections | /api/<collection> |
| Listing Sub-Collections | /api/<collection>/<id>/<sub-collection> |

| Querying Capability | Query Parameters |
|---------------------|-----------------------------------|
| Paging | offset, limit |
| Sorting | sort_by=attr, sort_order=asc desc |
| Filtering | filter[]="..." |

| Querying Capability | Query Parameters |
|----------------------|--|
| Querying by Tag | i.e. <code>by_tag=/department/finance</code> |
| Expanding Results | <code>expand=<what></code> , i.e. <code>expand=resources,tags,service_templates,...</code> |
| Selecting Attributes | <code>attributes=<attr1>,<attr2>,...</code> i.e. <code>attributes=id,name,type,...</code> |
| | Attributes can be: |
| | Database columns |
| | Virtual attributes |
| | Relationships |

2.4. COLLECTION QUERIES

| Collection | URL |
|---------------------|---------------------------------------|
| Automation Requests | <code>/api/automation_requests</code> |
| Availability Zones | <code>/api/availability_zones</code> |
| Clusters | <code>/api/clusters</code> |
| Conditions | <code>/api/conditions</code> |
| Datastores | <code>/api/data_stores</code> |
| Events | <code>/api/events</code> |
| Flavors | <code>/api/flavors</code> |
| Groups | <code>/api/groups</code> |
| Hosts | <code>/api/hosts</code> |
| Policies | <code>/api/policies</code> |
| Actions | <code>/api/policy_actions</code> |
| Policy Profiles | <code>/api/policy_profiles</code> |

| Collection | URL |
|--------------------|-------------------------|
| Providers | /api/providers |
| Provision Requests | /api/provision_requests |
| Request Tasks | /api/request_tasks |
| Requests | /api/requests |
| Resource Pools | /api/resource_pools |
| Roles | /api/roles |
| Security Groups | /api/security_groups |
| EVM Servers | /api/servers |
| Service Catalogs | /api/service_catalogs |
| Service Requests | /api/service_requests |
| Service Templates | /api/service_templates |
| Services | /api/services |
| Tags | /api/tags |
| Tasks | /api/tasks |
| Templates | /api/templates |
| Users | /api/users |
| Vms | /api/vms |
| Zones | /api/zones |

For example queries, see "Queries".

2.5. SUB-COLLECTION QUERIES

| Sub-Collection | URL |
|----------------|----------------------------|
| Tagging | /api/<collection>/:id/tags |

| Sub-Collection | URL |
|--|---|
| Policies | /api/<collection>/:id/policies |
| Policy Profiles | /api/<collection>/:id/policy_profiles |
| | |
| Service Requests | /api/service_templates/:id/service_requests |
| Request Tasks | |
| | /api/service_requests/:id/request_tasks |
| | /api/automation_requests/:id/request_tasks |
| | /api/provision_requests/:id/request_tasks |
| Request Tasks can also be accessed via the tasks alias | |
| | /api/service_requests/:id/tasks |
| | /api/automation_requests/:id/tasks |
| | /api/provision_requests/:id/tasks |

2.6. AVAILABLE ACTIONS

| Action | Method | URL | Example |
|-----------------------|--------|---|---------|
| Add Service Catalog | POST | /api/service_catalogs | |
| Add Service Catalogs | POST | /api/service_catalogs | |
| Edit Service Catalog | POST | /api/service_catalogs/<replaceable>id</replaceable> | |
| Edit Service Catalogs | POST | /api/service_catalogs | |
| Automation Request | POST | /api/automation_requests | |
| Automation Requests | POST | /api/automation_requests | |

| Action | Method | URL | Example |
|-----------------------------------|--------|--|---------|
| Edit Service | POST | /api/services/<replaceable>id</replaceable> | |
| Edit Service via PUT | PUT | /api/services/<replaceable>id</replaceable> | |
| Edit Service via PATCH | PATCH | /api/services/<replaceable>id</replaceable> | |
| Edit Services | POST | /api/services/ | |
| Assign Tags to a Service | POST | /api/services/<replaceable>id</replaceable>/tags | |
| Assign a Tag by Name to a Service | POST | /api/services/<replaceable>id</replaceable>/tags | |
| Assign a Tag by Name to a Service | POST | /api/services/<replaceable>id</replaceable>/tags | |
| Unassign Tags from Service | POST | /api/services/<replaceable>id</replaceable>/tags | |
| Retire Service Now | POST | /api/services/<replaceable>id</replaceable> | |
| Retire Service in Future | POST | /api/services/<replaceable>id</replaceable> | |
| Retire Services | POST | /api/services | |
| Delete Service | DELETE | /api/services/<replaceable>id</replaceable> | |
| Delete Services | POST | /api/services | |
| Edit Service Template | POST | /api/service_templates/<replaceable>id</replaceable> | |
| Edit Service Templates | POST | /api/service_templates | |

| Action | Method | URL | Example |
|-------------------------------------|--------|---|---------|
| Assign Tags to Service Template | POST | /api/service_templates/<replaceable>id</replaceable>/tags | |
| Unassign Tags from Service Template | POST | /api/service_templates/<replaceable>id</replaceable>/tags | |
| Delete Service Template | DELETE | /api/service_templates/<replaceable>id</replaceable> | |
| Delete Service Templates | POST | /api/service_templates | |
| Assign Service Templates | POST | /api/service_catalogs/<replaceable>id</replaceable>/service_templates | |
| Unassign Service Templates | POST | /api/service_catalogs/<replaceable>id</replaceable>/service_templates | |
| Order Service | POST | /api/service_catalogs/<replaceable>id</replaceable>/service_templates | |
| Order Services | POST | /api/service_catalogs/<replaceable>id</replaceable>/service_templates | |
| Delete Service Catalog | DELETE | /api/service_catalogs/<replaceable>id</replaceable> | |
| Delete Service Catalogs | POST | /api/service_catalogs | |
| Provision Request | POST | /api/provision_requests | |
| Provision Requests | POST | /api/provision_requests | |
| Create a Provider | POST | /api/providers | |

| Action | Method | URL | Example |
|---|--------|----------------|---------|
| Create a Provider with Compound Credentials | POST | /api/providers | |
| Edit a Provider | POST | /api/providers | |
| Update a Provider | POST | /api/providers | |
| Delete a Provider | POST | /api/providers | |
| Delete Multiple Providers | POST | /api/providers | |
| Refresh a Provider | POST | /api/providers | |
| Scan a VM | POST | /api/vms | |
| Set Owner of a VM | POST | /api/vms | |
| Add a Lifecycle Event to a VM | POST | /api/vms | |
| Add an Event to a VM | POST | /api/vms | |
| Start a VM | POST | /api/vms | |
| Stop a VM | POST | /api/vms | |
| Suspend a VM | POST | /api/vms | |
| Delete VMs | DELETE | /api/vms | |

2.7. PROVISIONING REQUEST ATTRIBUTES

- [Request Attribute Groups](#)
- [Requester Attributes](#)
- [Custom Attributes](#)
- [Environment Attributes](#)
- [Service Catalog Attributes](#)
- [Schedule Attributes](#)
- [Network Attributes](#)

- [Hardware Attributes](#)

2.7.1. Provisioning Request Attribute Groups

| Attribute Group | Type | Description |
|-------------------|--------|--|
| version | string | Interface version. Defaults to 1.1 |
| template_fields | hash | <p>Fields used to find the template virtual machine. Provide any or all fields. Supply a guid or <code>ems_guid</code> to protect against matching same-named templates on different Providers within the appliance.</p> <p>Supported fields are: <code>name=[VM Template Name]</code> Example: <code>template_1 guid=[guid value from vms resource] ems_guid=[uid_ems value from vms resource]</code></p> |
| vm_fields | hash | Allows for setting properties on the Service Catalog, Hardware, Network, Customize, and Schedule tabs in the Provisioning dialog. |
| requester | hash | Allows for the setting of properties on the Requester tab in the Provisioning dialog. |
| tags | hash | <p>Tags to apply to newly created virtual machine.</p> <p>Example: <code>network_location=Internal</code> <code>cc=001</code></p> |
| additional_values | hash | <p>Additional values are name-value pairs stored with a provision request, but not used by the core provisioning code. These values are usually referenced from automate methods for custom processing.</p> <p>Example: Store a <code>request_id</code> from an external system so the system can be notified during the provisioning process.</p> |

| Attribute Group | Type | Description |
|-----------------------|------|--|
| ems_custom_attributes | hash | Custom attributes applied to the virtual machine through the Provider as part of provisioning. |
| miq_custom_attributes | hash | Custom attributes applied to the virtual machine and stored in the database as part of provisioning. |

2.7.2. Service Catalog Attributes

These attributes are used in the `vm_fields` attribute group:

| Attribute | Type | Description | Default |
|----------------|---------|--|---------|
| number_of_vms | integer | Number of virtual machines - maximum 50 | 1 |
| vm_description | string | Virtual machine description - maximum 100 characters | |
| vm_prefix | string | Virtual machine name prefix or suffix | |
| src_vm_id | integer | Name | |
| vm_name | string | Virtual machine name | |
| pxe_image_id | string | Image | |
| pxe_server_id | integer | Server | |
| host_name | string | Host name | |
| provision_type | string | Provision type | vmware |
| linked_clone | boolean | Linked clone | false |
| snapshot | string | Snapshot | |
| vm_filter | integer | Filter | |

2.7.3. Hardware Attributes

These attributes are used in the `vm_fields` attribute group:

| Attribute | Type | Description | Values | Default |
|-------------------|---------|-------------------|------------------------|-----------|
| disk_format | string | Disk format | thick, thin, unchanged | unchanged |
| cpu_limit | integer | CPU (MHz) | | |
| memory_limit | integer | Memory (MB) | | |
| number_of_sockets | integer | Number of sockets | 1, 2, 4, 8 | 1 |
| cores_per_socket | integer | Cores per socket | 1, 2, 4, 8 | 1 |
| cpu_reserve | integer | CPU (MHz) | | |
| vm_memory | string | Memory (MB) | 1024, 2048, 4096 | 1024 |
| memory_reserve | integer | Memory (MB) | | |
| network_adapters | integer | Network adapters | 1, 2, 3, 4 | 1 |

2.7.4. Network Attributes

These attributes are used in the `vm_fields` attribute group:

| Attribute | Type | Description |
|-------------|--------|-------------|
| vlan | string | vLAN |
| mac_address | string | MAC address |

2.7.5. Custom Attributes

These attributes are used in the `vm_fields` attribute group:

| Attribute | Type | Description | Values | Default |
|----------------------|--------|----------------------------|--------|---------|
| dns_servers | string | DNS server list | | |
| sysprep_organization | string | Organization | | |
| sysprep_password | string | New administrator password | | |

| Attribute | Type | Description | Values | Default |
|------------------------------------|---------|------------------------|--------------------|-----------|
| sysprep_custom_spec | string | Name | | |
| sysprep_server_license_mode | string | Identification | perServer, perSeat | |
| ldap_ous | string | LDAP group | | |
| sysprep_timezone | string | Timezone | | |
| dns_suffixes | string | DNS suffix list | | |
| sysprep_product_id | string | ProductID | | |
| sysprep_identification | string | Identification | domain, workgroup | |
| sysprep_per_server_max_connections | string | Maximum connections | | 5 |
| sysprep_computer_name | string | Computer name | | |
| sysprep_workgroup_name | string | Workgroup name | | WORKGROUP |
| sysprep_spec_override | boolean | Override specification | | false |
| addr_mode | string | Address mode | static, dhcp | dhcp |
| linux_host_name | string | Computer name | | |
| sysprep_domain_admin | string | Domain admin | | |
| sysprep_change_sid | boolean | Change SID | | true |
| sysprep_domain_name | string | Domain name | | |
| sysprep_upload_file | string | Upload | | |

| Attribute | Type | Description | Values | Default |
|-------------------------------|---------|------------------|---------|----------|
| gateway | string | Gateway | | |
| ip_addr | string | IP address | | |
| linux_domain_name | string | Domain name | | |
| sysprep_domain_password | string | Domain password | | |
| sysprep_auto_logon | boolean | Auto Logon | | true |
| sysprep_enabled | string | Customize | | disabled |
| sysprep_delete_accounts | boolean | Delete accounts | | false |
| sysprep_upload_text | string | Sysprep text | | |
| wins_servers | string | WINS server list | | |
| subnet_mask | string | Subnet mask | | |
| sysprep_full_name | string | Full name | | |
| sysprep_auto_logon_count | integer | Auto logon count | 1, 2, 3 | 1 |
| customization_template_id | integer | Script name | | |
| root_password | string | Root password | | |
| hostname | string | Host name | | |
| customization_template_script | string | Script text | | |

2.7.6. Schedule Attributes

These attributes are used in the `vm_fields` attribute group:

| Attribute | Type | Description | Values | Default |
|-----------------|---------|--|--|-------------|
| schedule_type | string | When to provision | schedule, immediately (<i>On Approval</i>) | immediately |
| vm_auto_start | boolean | Power on virtual machines after creation | | true |
| schedule_time | time | Time to provision on | | |
| retirement | integer | Time until retirement | 0 (<i>Indefinite</i>), 1.month, 3.months, 6.months | 0 |
| retirement_warn | integer | Retirement warning | 1.week, 2.weeks, 30.days | 1.week |

2.7.7. Requester Attributes

These attributes are used in the **requester** attribute group:

| Attribute | Type | Description |
|---------------------|--------|------------------------|
| owner_phone | string | Phone |
| owner_country | string | Country/Region |
| owner_phone_mobile | string | Mobile phone |
| owner_title | string | Title |
| owner_first_name | string | First name |
| owner_manager | string | Manager name |
| owner_address | string | Address |
| owner_company | string | Company |
| owner_last_name | string | Last name |
| owner_manager_email | string | Manager e-mail address |

| Attribute | Type | Description |
|---------------------|--------|-----------------------------|
| owner_city | string | City |
| owner_department | string | Department |
| owner_load_ldap | button | Look up LDAP e-mail address |
| owner_manager_phone | string | Manager phone |
| owner_state | string | State |
| owner_office | string | Office |
| owner_zip | string | Zip code |
| owner_email | string | E-Mail |
| request_notes | string | Notes |

2.7.8. Environment Attributes

These attributes cannot be passed directly. To use these attributes, provide **additional_values** and allow customization methods to use these attributes, then modify the request accordingly.

| Attribute | Type | Description | Values | Default |
|----------------------------------|---------|---------------------|--------|---------|
| new_datastore_grow_increment | integer | Grow increment (GB) | | |
| new_datastore_create | boolean | Create datastore | | false |
| placement_cluster_name | integer | Name | | |
| new_datastore_aggregate | string | Aggregate | | |
| new_datastore_max_size | integer | Max size (GB) | | |
| new_datastore_storage_controller | string | Controller | | |
| cluster_filter | integer | Filter | | |

| Attribute | Type | Description | Values | Default |
|----------------------------------|---------|-------------------------|-----------|---------|
| host_filter | integer | Filter | | |
| ds_filter | integer | Filter | | |
| new_datastore_vo lume | string | Volume | | |
| placement_host_n ame | integer | Name | | |
| placement_ds_na me | integer | Name | | |
| new_datastore_fs_ type | string | FS Type | NFS, VMFS | NFS |
| rp_filter | integer | Filter | | |
| new_datastore_thi n_provision | string | Thin provision | | |
| placement_auto | boolean | Choose automatically | | false |
| new_datastore_siz e | integer | Size (GB) | | |
| new_datastore_au togrow | string | Autogrow | | false |
| placement_folder_ name | integer | Name | | |
| new_datastore_na me | string | Name | | |
| placement_rp_na me | integer | Name | | |
| placement_dc_na me | integer | Name | | |

CHAPTER 3. EXAMPLES

This section provides a collection of examples of using the REST API to interact with resources in a Red Hat CloudForms environment.

3.1. GENERAL QUERIES

This section introduces a number of general examples of how to use the REST API to query resources, and return information about resources and events.

3.1.1. Queries

Query all virtual machines:

```
GET /api/vms
```

Query a specific virtual machine:

```
GET /api/vms/1386
```

Query all virtual machines, but return only the name and vendor:

```
GET /api/vms?expand=resources&attributes=name,vendor
```

Query all virtual machines named sample*, and return the name and vendor:

```
GET /api/vms?  
expand=resources&attributes=name,vendor&filter[]="name='sample%'"
```

Query all virtual machines, but only return the first 500 results:

```
GET /api/vms?offset=0&limit=500
```

Query all virtual machines, but return the second 500 results:

```
GET /api/vms?offset=500&limit=500
```

Query the first 1000 virtual machines named test*, get the name, vendor, and guid, and sort by name in ascending order:

```
GET /api/vms?  
offset=0&limit=1000&filter[]="name='test%'"&expand=resources&attributes=na  
me,vendor,guid&sort_by=name&sort_order=asc
```

Query services tagged for the finance department:

```
GET /api/services?by_tag=/department/finance
```

Get the details of tags for the first service:

```
GET /api/services/1/tags?expand=resources
```


Get the details of the first service catalog and related details on the assigned service templates:

```
GET /api/service_catalogs/1?expand=service_templates
```

```
GET /api/service_templates/25/service_requests?
expand=resources,request_tasks
```

Get a specific provision request with expanded details on the associated provision request tasks:

```
GET /api/provision_requests/120?expand=tasks
```

3.1.2. Paging Queries

This series of requests shows paging queries and the expected responses for each subsequent page.

3.1.2.1. Request:

```
GET /api/vms?offset=0&limit=500
&sort_order=asc&sort_by=name
&expand=resources&attributes=name
```

3.1.2.2. Response:

```
{
  "name": "vms",
  "count": 1912,
  "subcount": 500,
  "resources": [
    {
      "href": "https://hostname/api/vms/176",
      "id": 176,
      "name": "53 Zone1"
    },
    ...
    {
      "href": "https://hostname/api/vms/1575",
      "id": 1575,
      "name": "VmEmpty-3de98f0f-c6f3-4f8b-a932-554713a61067"
    }
  ],
  "actions": [
  ]
}
```

3.1.2.3. Request:

```
GET /api/vms?offset=500&limit=500
&sort_order=asc&sort_by=name
&expand=resources&attributes=name
```

3.1.2.4. Response:

```

{
  "name": "vms",
  "count": 1912,
  "subcount": 500,
  "resources": [
    {
      "href": "https://hostname/api/vms/1574",
      "id": 1574,
      "name": "VmEmpty-3e13ff43-6907-4a22-8f95-58aeb1bffa0b"
    },
    ...
    {
      "href": "https://hostname/api/vms/1076",
      "id": 1076,
      "name": "VmEmpty-9a885181-7771-4f91-9805-245c7606d833"
    }
  ],
  "actions": [
  ]
}

```

3.1.2.5. Request:

```

GET /api/vms?offset=1000&limit=500
    &sort_order=asc&sort_by=name
    &expand=resources&attributes=name

```

3.1.2.6. Response:

```

{
  "name": "vms",
  "count": 1912,
  "subcount": 500,
  "resources": [
    {
      "href": "https://hostname/api/vms/1074",
      "id": 1074,
      "name": "VmEmpty-9ab9e101-92b0-4b6b-864e-e196538da8a8"
    },
    ...
    {
      "href": "https://hostname/api/vms/575",
      "id": 575,
      "name": "VmEmpty-f251f135-01c8-4d44-b8e1-37b30844a9dd"
    }
  ],
  "actions": [
  ]
}

```

3.1.2.7. Request:

```
GET /api/vms?offset=1500&limit=500
    &sort_order=asc&sort_by=name
    &expand=resources&attributes=name
```

3.1.2.8. Response:

```
{
  "name": "vms",
  "count": 1912,
  "subcount": 412,
  "resources": [
    {
      "href": "https://hostname/api/vms/574",
      "id": 574,
      "name": "VmEmpty-f28912f3-b096-487f-9763-97b39b67364b"
    },
    ...
    {
      "href": "https://hostname/api/vms/1907",
      "id": 1907,
      "name": "yy_vm"
    }
  ],
  "actions": [
  ]
}
```



NOTE

In this last request, the **subcount** was less than the requested page size, thus denoting the last page of data being returned.

3.1.3. Querying a Delete Task

3.1.3.1. Request:

```
GET /api/tasks/625
```

3.1.3.2. Response:

```
{
  "href": "https://hostname/api/tasks/625",
  "id": 625,
  "name": "Provider id:106 name:'rhevm102' deleting",
  "state": "Finished",
  "status": "Ok",
  "message": "Task completed successfully",
  "userid": "admin",
  "created_on": "2015-05-06T14:02:26Z",
  "updated_on": "2015-05-06T14:02:32Z"
}
```

3.2. SERVICE CATALOGS

This section provides examples of how to interact with service catalogs.

3.2.1. Adding a Sample Service Catalog

3.2.1.1. Request:

```
POST /api/service_catalogs
```

```
{
  "action" : "create",
  "resource" : {
    "name" : "Sample Service Catalog",
    "description" : "Description of Sample Service Catalog",
    "service_templates" : [
      { "href" : "https://hostname/api/service_templates/3" },
      { "href" : "https://hostname/api/service_templates/4" }
    ]
  }
}
```

3.2.1.2. Response:

```
{
  "results": [
    {
      "id": 7,
      "name": "Sample Service Catalog",
      "description": "Description of Sample Service Catalog"
    }
  ]
}
```

3.2.2. Adding Multiple Service Catalogs

3.2.2.1. Request:

```
POST /api/service_catalogs
```

```
{
  "action" : "create",
  "resources" : [
    {
      "name" : "First Sample Service Catalog",
      "description" : "Description of First Sample Service Catalog",
      "service_templates" : [
        { "href" : "https://hostname/api/service_templates/1" },
        { "href" : "https://hostname/api/service_templates/2" }
      ]
    },
  ],
}
```

```

{
  "name" : "Second Sample Service Catalog",
  "description" : "Description of Second Sample Service Catalog",
  "service_templates" : [
    { "href" : "https://hostname/api/service_templates/3" },
    { "href" : "https://hostname/api/service_templates/4" }
  ]
},
{
  "name" : "Third Sample Service Catalog",
  "description" : "Description of Third Sample Service Catalog",
  "service_templates" : [
    { "href" : "https://hostname/api/service_templates/5" },
    { "href" : "https://hostname/api/service_templates/6" }
  ]
}
]
}

```

3.2.2.2. Response:

```

{
  "results": [
    {
      "id": 8,
      "name": "First Sample Service Catalog",
      "description": "Description of First Sample Service Catalog"
    },
    {
      "id": 9,
      "name": "Second Sample Service Catalog",
      "description": "Description of Second Sample Service Catalog"
    },
    {
      "id": 10,
      "name": "Third Sample Service Catalog",
      "description": "Description of Third Sample Service Catalog"
    }
  ]
}

```

3.2.3. Assigning Service Templates to Service Catalogs

3.2.3.1. Request:

```
POST /api/service_catalogs/6/service_templates
```

```

{
  "action" : "assign",
  "resources" : [
    { "href" : "https://hostname/api/service_templates/5" },

```

```

    { "href" : "https://hostname/api/service_templates/1" }
  ]
}

```

3.2.3.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Service Template id:5 name:'template5'",
      "service_template_id": 5,
      "service_template_href": "https://hostname/api/service_templates/5",
      "href": "https://hostname/api/service_catalogs/6"
    },
    {
      "success": false,
      "message": "Service Template 1 is currently assigned to Service
Catalog 3",
      "service_template_id": 1,
      "service_template_href": "https://hostname/api/service_templates/1",
      "href": "https://hostname/api/service_catalogs/6"
    }
  ]
}

```

3.2.4. Editing a Service Catalog

3.2.4.1. Request:

```
POST /api/service_catalog/7
```

```

{
  "action" : "edit",
  "resource" : {
    "description" : "Updated Description of the Seventh Service Catalog"
  }
}

```

3.2.4.2. Response:

```

{
  "href": "https://hostname/api/service_catalogs/7",
  "id": 7,
  "name": "Sample Service Catalog",
  "description": "Updated Description of the Seventh Service Catalog",
  "service_templates": {
    "count": 1,
    "resources": [
      {
        "href":
"https://hostname/api/service_catalogs/7/service_templates/3"

```

```

    }
  ]
}

```

3.2.5. Editing Multiple Service Catalogs

3.2.5.1. Request:

```
POST /api/service_catalogs
```

```

{
  "action" : "edit",
  "resources" : [
    {
      "href" : "https://hostname/api/service_catalogs/3",
      "description" : "Updated Description for Third Service Catalog"
    },
    {
      "href" : "https://hostname/api/service_catalogs/6",
      "description" : "Updated Description for Sixth Service Catalog"
    }
  ]
}

```

3.2.5.2. Response:

```

{
  "results": [
    {
      "id": 3,
      "name": "Service Catalog B Added from REST API",
      "description": "Updated Description for Third Service Catalog"
    },
    {
      "id": 6,
      "name": "Service Catalog E Added from REST API",
      "description": "Updated Description for Sixth Service Catalog"
    }
  ]
}

```

3.2.6. Ordering a Single Service from a Service Catalog

3.2.6.1. Request:

```
POST /api/service_catalogs/1/service_templates
```

```

{
  "action" : "order",
  "resource" : {

```

```

    "href" : "https://hostname/api/service_templates/3",
    "option_0_vm_target_name" : "test-vm-0001",
    "option_0_vm_target_hostname" : "test-vm-0001"
  }
}

```

3.2.6.2. Response:

```

{
  results: [
    {
      "approval_state": "pending_approval",
      "created_on": "2014-07-02T19:28:12Z",
      "description": "Provisioning Service [aws-ubuntu-api] from [aws-ubuntu-api]",
      "destination_id": null,
      "destination_type": null,
      "fulfilled_on": null,
      "id": 13,
      "message": "Service_Template_Provisioning - Request Created",
      "options": {
        "dialog": {
          "dialog_option_0_vm_target_name" : "test-vm-0001",
          "dialog_option_0_vm_target_hostname" : "test-vm-0001"
        }
      }
      "request_state": "pending",
      "request_type": "clone_to_service",
      "requester_id": 1,
      "requester_name": "Administrator",
      "source_id": 6,
      "source_type": "ServiceTemplate",
      "status": "Ok",
      "updated_on": "2014-07-02T19:28:12Z",
      "userid": "admin"
    }
  ]
}

```

3.2.7. Ordering Multiple Services from a Service Catalog

3.2.7.1. Request:

```
POST /api/service_catalogs/2/service_templates
```

```

{
  "action" : "order",
  "resources" : [
    {
      "href" : "https://hostname/api/service_templates/3",
      "option_1_vm_target_name" : "sample-vm-1201",
      "option_2_vm_target_hostname" : "sample-vm-1201"
    },
  ],
}

```



```

{
  "href" : "https://hostname/api/service_templates/3",
  "option_1_vm_target_name" : "sample-vm-1202",
  "option_2_vm_target_hostname" : "sample-vm-1202"
},
{
  "href" : "https://hostname/api/service_templates/4",
  "option_1_vm_target_name" : "dev-vm1",
  "option_2_vm_target_hostname" : "dev-vm1",
  "option_3_vm_memory" : '16384'
},
]
}

```

3.2.7.2. Response:

```

{
  results: [
    {
      "approval_state": "pending_approval",
      "created_on": "2014-07-02T19:28:12Z",
      "description": "Provisioning Service [sample-vm-1201] from [sample-vm-1201]",
      "destination_id": null,
      "destination_type": null,
      "fulfilled_on": null,
      "id": 13,
      "message": "Service_Template_Provisioning - Request Created",
      "options": {
        "dialog": {
          "dialog_option_0_vm_target_name" : "test-vm-0001",
          "dialog_option_0_vm_target_hostname" : "test-vm-0001"
        }
      }
    },
    {
      "request_state": "pending",
      "request_type": "clone_to_service",
      "requester_id": 1,
      "requester_name": "Administrator",
      "source_id": 6,
      "source_type": "ServiceTemplate",
      "status": "Ok",
      "updated_on": "2014-07-02T19:28:12Z",
      "userid": "admin"
    },
    {
      "approval_state": "pending_approval",
      "created_on": "2014-07-02T19:28:12Z",
      "description": "Provisioning Service [sample-vm-1202] from [sample-vm-1202]",
      "destination_id": null,
      "destination_type": null,
      "fulfilled_on": null,
      "id": 13,
      "message": "Service_Template_Provisioning - Request Created",
      "options": {

```

```

        "dialog": {
          "dialog_option_0_vm_target_name" : "test-vm-0001",
          "dialog_option_0_vm_target_hostname" : "test-vm-0001"
        }
      }
    "request_state": "pending",
    "request_type": "clone_to_service",
    "requester_id": 1,
    "requester_name": "Administrator",
    "source_id": 6,
    "source_type": "ServiceTemplate",
    "status": "Ok",
    "updated_on": "2014-07-02T19:28:12Z",
    "userid": "admin"
  },
  {
    "approval_state": "pending_approval",
    "created_on": "2014-07-02T19:28:12Z",
    "description": "Provisioning Service [sample-vm-1201] from [sample-vm-1201]",
    "destination_id": null,
    "destination_type": null,
    "fulfilled_on": null,
    "id": 13,
    "message": "Service_Template_Provisioning - Request Created",
    "options": {
      "dialog": {
        "dialog_option_0_vm_target_name" : "test-vm-0001",
        "dialog_option_0_vm_target_hostname" : "test-vm-0001"
      }
    }
  }
  "request_state": "pending",
  "request_type": "clone_to_service",
  "requester_id": 1,
  "requester_name": "Administrator",
  "source_id": 6,
  "source_type": "ServiceTemplate",
  "status": "Ok",
  "updated_on": "2014-07-02T19:28:12Z",
  "userid": "admin"
}
]
}

```

3.2.8. Unassigning Service Templates from a Service Catalog

3.2.8.1. Request:

```
POST /api/service_catalogs/6/service_templates
```

```

{
  "action" : "unassign",
  "resources" : [
    { "href" : "https://hostname/api/service_templates/1" },
  ]
}

```

```

    { "href" : "https://hostname/api/service_templates/5" },
    { "href" : "https://hostname/api/service_templates/8" }
  ]
}

```

3.2.8.2. Response:

```

{
  "results": [
    {
      "success": false,
      "message": "Service Template 1 is not currently assigned to Service Catalog 3",
      "service_template_id": 1,
      "service_template_href": "https://hostname/api/service_templates/1",
      "href": "https://hostname/api/service_catalogs/6"
    },
    {
      "success": true,
      "message": "Unassigning Service Template id:5 name:'template5'",
      "service_template_id": 5,
      "service_template_href": "https://hostname/api/service_templates/5",
      "href": "https://hostname/api/service_catalogs/6"
    },
    {
      "success": false,
      "message": "Couldn't find ServiceTemplate with id=8",
      "href": "https://hostname/api/service_catalogs/6"
    }
  ]
}

```

3.2.9. Deleting Multiple Service Catalogs

3.2.9.1. Request:

```
POST /api/service_catalogs
```

```

{
  "action" : "delete",
  "resources" : [
    { "href" : "https://hostname/api/service_catalogs/8" },
    { "href" : "https://hostname/api/service_catalogs/9" },
    { "href" : "https://hostname/api/service_catalogs/10" }
  ]
}

```

3.2.9.2. Response:

```

{
  "results": [
    {

```

```

    "success": true,
    "message": "service_catalogs id: 8 deleting",
    "href": "https://hostname/api/service_catalogs/8"
  },
  {
    "success": true,
    "message": "service_catalogs id: 9 deleting",
    "href": "https://hostname/api/service_catalogs/9"
  },
  {
    "success": true,
    "message": "service_catalogs id: 10 deleting",
    "href": "https://hostname/api/service_catalogs/10"
  }
]
}

```

3.3. TAGS

This section provides examples of how to interact with tags.

3.3.1. Assigning Tags to a Service

3.3.1.1. Request:

```
POST /api/services/101/tags
```

```

{
  "action" : "assign",
  "resources" : [
    { "category" : "location", "name" : "ny" },
    { "category" : "department", "name" : "finance" },
    { "category" : "environment", "name" : "dev" }
  ]
}

```

3.3.1.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/services/101",
      "tag_category": "location",
      "tag_name": "ny"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance"
    }
  ]
}

```

```

    },
    {
      "success": true,
      "message": "Assigning Tag: category:'environment' name:'dev'",
      "href": "https://hostname/api/services/101",
      "tag_category": "environment",
      "tag_name": "dev"
    }
  ]
}

```

3.3.2. Assigning Tags by Name to a Service

3.3.2.1. Request:

```
POST /api/services/101/tags
```

```

{
  "action" : "assign",
  "resources" : [
    { "name" : "/department/finance" },
    { "name" : "/location/ny" }
  ]
}

```

3.3.2.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/services/101",
      "tag_category": "location",
      "tag_name": "ny"
    }
  ]
}

```

3.3.3. Assigning Tags by Reference to a Service

3.3.3.1. Request:

```
POST /api/services/101/tags
```

```
{
  "action" : "assign",
  "resources" : [
    { "href" : "https://hostname/api/services/1/tags/49" }
  ]
}
```

3.3.3.2. Response:

```
{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance",
      "tag_href": "https://hostname/api/tags/49"
    }
  ]
}
```

3.3.4. Assigning Tags to a Service Template

3.3.4.1. Request:

POST /api/service_templates/1/tags

```
{
  "action" : "assign",
  "resources" : [
    { "category" : "location", "name" : "ny" },
    { "category" : "department", "name" : "finance" }
  ]
}
```

3.3.4.2. Response:

```
{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/service_templates/1",
      "tag_category": "location",
      "tag_name": "ny"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/service_templates/1",
      "tag_category": "department",
    }
  ]
}
```

```

    "tag_name": "finance"
  }
]
}

```

3.3.5. Assigning Tags to a Virtual Machine

3.3.5.1. Request:

```
POST /api/vms/101/tags
```

```

{
  "action" : "assign",
  "resources" : [
    { "name" : "/department/finance" },
    { "name" : "/location/ny" }
  ]
}

```

3.3.5.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "location",
      "tag_name": "ny"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "department",
      "tag_name": "finance"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'environment' name:'dev'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "environment",
      "tag_name": "dev"
    }
  ]
}

```

3.3.6. Assigning Tags by Name to a Virtual Machine

3.3.6.1. Request:

```
POST /api/vms/101/tags
```

```

{
  "action" : "assign",
  "resources" : [
    { "name" : "/department/finance" },
    { "name" : "/location/ny" }
  ]
}

```

3.3.6.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "department",
      "tag_name": "finance"
    },
    {
      "success": true,
      "message": "Assigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "location",
      "tag_name": "ny"
    }
  ]
}

```

3.3.7. Assigning Tags by Reference to a Virtual Machine

3.3.7.1. Request:

```
POST /api/vms/101/tags
```

```

{
  "action" : "assign",
  "resources" : [
    { "href" : "https://hostname/api/vms/1/tags/49" }
  ]
}

```

3.3.7.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Assigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/vms/101",
      "tag_category": "department",

```



```

    "tag_name": "finance",
    "tag_href": "https://hostname/api/tags/49"
  }
]
}

```

3.3.8. Unassigning Tags from a Service

3.3.8.1. Request:

```
POST /api/services/101/tags
```

```

{
  "action" : "unassign",
  "resources" : [
    { "category" : "department", "name" : "finance" },
    { "category" : "environment", "name" : "dev" }
  ]
}

```

3.3.8.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Unassigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance"
    },
    {
      "success": true,
      "message": "Unassigning Tag: category:'environment' name:'dev'",
      "href": "https://hostname/api/services/101",
      "tag_category": "environment",
      "tag_name": "dev"
    }
  ]
}

```

3.3.9. Unassigning a Tag by Name from a Service

3.3.9.1. Request:

```
POST /api/services/101/tags
```

```

{
  "action" : "unassign",
  "resources" : [

```

```
{
  "name" : "/managed/department/finance"
}
```

3.3.9.2. Response:

```
{
  "results": [
    {
      "success": true,
      "message": "Unassigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance"
    }
  ]
}
```

3.3.10. Unassigning a Tag by Reference from a Service

3.3.10.1. Request:

```
POST /api/services/101/tags
```

```
{
  "action" : "unassign",
  "resources" : [
    { "href" : "tags/49" }
  ]
}
```

3.3.10.2. Response:

```
{
  "results": [
    {
      "success": true,
      "message": "Unassigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/services/101",
      "tag_category": "department",
      "tag_name": "finance",
      "tag_href": "https://hostname/api/tags/49"
    }
  ]
}
```

3.3.11. Unassigning Tags from a Service Template

3.3.11.1. Request:

```
POST /api/service_templates/1/tags
```

```

{
  "action" : "unassign",
  "resources" : [
    { "category" : "location", "name" : "ny" },
    { "category" : "department", "name" : "finance" }
  ]
}

```

3.3.11.2. Response:

```

{
  "results": [
    {
      "success": true,
      "message": "Unassigning Tag: category:'location' name:'ny'",
      "href": "https://hostname/api/service_templates/1",
      "tag_category": "location",
      "tag_name": "ny"
    },
    {
      "success": true,
      "message": "Unassigning Tag: category:'department' name:'finance'",
      "href": "https://hostname/api/service_templates/1",
      "tag_category": "department",
      "tag_name": "finance"
    }
  ]
}

```

3.4. AUTOMATION REQUESTS

This section provides examples of how to interact with automation requests.

3.4.1. Triggering a Single Automation Request

With automation requests:

- **version** defaults to **1.1** if not specified.
- **user_name** defaults to the **REST API** authenticated user if not specified.

3.4.1.1. Request:

POST /api/automation_requests

```

{
  "version" : "1.1",
  "uri_parts" : {
    "namespace" : "System",
    "class" : "Request",
    "instance" : "InspectME",

```

```

    "message" : "create"
  },
  "parameters" : {
    "var1" : "xxxxx",
    "var2" : "yyyyy",
    "var3" : 1024,
    "var4" : true,
    "var5" : "last value"
  },
  "requester" : {
    "user_name" : "admin",
    "auto_approve" : true
  }
}

```

Optionally, the action-based request format is also supported:

3.4.1.2. Request:

POST /api/automation_requests

```

{
  "action" : "create",
  "resource" : {
    "version" : "1.1",
    "uri_parts" : {
      "namespace" : "System",
      "class" : "Request",
      "instance" : "InspectME",
      "message" : "create"
    }
  },
  "parameters" : {
    "var1" : "xxxxx",
    "var2" : "yyyyy",
    "var3" : 1024,
    "var4" : true,
    "var5" : "last value"
  },
  "requester" : {
    "user_name" : "admin",
    "auto_approve" : true
  }
}

```

3.4.1.3. Response:

```

{
  "results": [
    {
      "id": 12,
      "description": "Automation Task",
      "approval_state": "approved",
      "type": "AutomationRequest",

```

```

    "created_on": "2015-04-16T21:49:55Z",
    "updated_on": "2015-04-16T21:49:55Z",
    "requester_id": 1,
    "requester_name": "Administrator",
    "request_type": "automation",
    "request_state": "pending",
    "status": "Ok",
    "options": {
      "message": "create",
      "namespace": "System",
      "class_name": "Request",
      "instance_name": "InspectME",
      "user_id": 1,
      "attrs": {
        "var1": "xxxxx",
        "var2": "yyyyy",
        "var3": 1024,
        "var4": true,
        "var5": "last value",
        "userid": "admin"
      }
    },
    "userid": "admin"
  }
]
}

```

3.4.2. Triggering Multiple Automation Requests

With automation requests:

- **version** defaults to **1.1** if not specified.
- **user_name** defaults to the **REST API** authenticated user if not specified.

3.4.2.1. Request:

```
POST /api/automation_requests
```

```

{
  "action" : "create",
  "resources" : [
    {
      "version" : "1.1",
      "uri_parts" : {
        "namespace" : "System",
        "class" : "Request",
        "instance" : "InspectME",
        "message" : "create"
      },
      "parameters" : {
        "vm_name" : "test_1",
        "var2" : "yyyyy",
        "var3" : 1024,
        "var4" : true,

```

```
    "var5" : "last value"
  },
  "requester" : {
    "user_name" : "jdoe",
    "auto_approve" : true
  }
},
{
  "uri_parts" : {
    "namespace" : "System",
    "class" : "Request",
    "instance" : "InspectME",
    "message" : "create"
  },
  "parameters" : {
    "vm_name" : "test_2",
    "vm_memory" : 1024,
    "memory_limit" : 16384
  },
  "requester" : {
    "auto_approve" : true
  }
},
{
  "uri_parts" : {
    "namespace" : "System",
    "class" : "Request",
    "instance" : "InspectME",
    "message" : "create"
  },
  "parameters" : {
    "vm_name" : "test_3",
    "vm_memory" : 2048,
    "memory_limit" : 16384
  },
  "requester" : {
    "auto_approve" : true
  }
},
{
  "uri_parts" : {
    "namespace" : "System",
    "class" : "Request",
    "instance" : "InspectME",
    "message" : "create"
  },
  "parameters" : {
    "vm_name" : "test_4",
    "vm_memory" : 4096,
    "memory_limit" : 16384
  },
  "requester" : {
    "auto_approve" : true
  }
}
```

```

    }
  ]
}

```

3.4.2.2. Response:

```

{
  "results": [
    {
      "id": 14,
      "description": "Automation Task",
      "approval_state": "approved",
      "type": "AutomationRequest",
      "created_on": "2015-04-16T21:59:42Z",
      "updated_on": "2015-04-16T21:59:42Z",
      "requester_id": 13,
      "requester_name": "aab",
      "request_type": "automation",
      "request_state": "pending",
      "status": "Ok",
      "options": {
        "message": "create",
        "namespace": "System",
        "class_name": "Request",
        "instance_name": "InspectME",
        "user_id": 13,
        "attrs": {
          "vm_name": "test_1",
          "var2": "yyyyy",
          "var3": 1024,
          "var4": true,
          "var5": "last value",
          "userid": "aab"
        }
      }
    },
    {
      "userid": "aab"
    }
  ],
  {
    {
      "id": 15,
      "description": "Automation Task",
      "approval_state": "approved",
      "type": "AutomationRequest",
      "created_on": "2015-04-16T21:59:42Z",
      "updated_on": "2015-04-16T21:59:42Z",
      "requester_id": 1,
      "requester_name": "Administrator",
      "request_type": "automation",
      "request_state": "pending",
      "status": "Ok",
      "options": {
        "message": "create",
        "namespace": "System",
        "class_name": "Request",
        "instance_name": "InspectME",
        "user_id": 1,

```

```
    "attrs": {
      "vm_name": "test_2",
      "vm_memory": 1024,
      "memory_limit": 16384,
      "userid": "admin"
    }
  },
  "userid": "admin"
},
{
  "id": 16,
  "description": "Automation Task",
  "approval_state": "approved",
  "type": "AutomationRequest",
  "created_on": "2015-04-16T21:59:42Z",
  "updated_on": "2015-04-16T21:59:42Z",
  "requester_id": 1,
  "requester_name": "Administrator",
  "request_type": "automation",
  "request_state": "pending",
  "status": "Ok",
  "options": {
    "message": "create",
    "namespace": "System",
    "class_name": "Request",
    "instance_name": "InspectME",
    "user_id": 1,
    "attrs": {
      "vm_name": "test_3",
      "vm_memory": 2048,
      "memory_limit": 16384,
      "userid": "admin"
    }
  }
},
"userid": "admin"
},
{
  "id": 17,
  "description": "Automation Task",
  "approval_state": "approved",
  "type": "AutomationRequest",
  "created_on": "2015-04-16T21:59:42Z",
  "updated_on": "2015-04-16T21:59:42Z",
  "requester_id": 1,
  "requester_name": "Administrator",
  "request_type": "automation",
  "request_state": "pending",
  "status": "Ok",
  "options": {
    "message": "create",
    "namespace": "System",
    "class_name": "Request",
    "instance_name": "InspectME",
    "user_id": 1,
    "attrs": {
      "vm_name": "test_4",
```



```

        "vm_memory": 4096,
        "memory_limit": 16384,
        "userid": "admin"
    }
},
"userid": "admin"
}
]
}

```

3.5. PROVISIONING REQUESTS

This section provides examples of how to interact with provisioning requests.

3.5.1. Triggering a Single Provision Request

With provisioning requests:

- **version** defaults to **1.1** if not specified.
- **user_name** defaults to the **REST API** authenticated user if not specified.

3.5.1.1. Request:

Provisioning requests are made available via the following endpoint, either by specifying a **create** action or by posting the request directly to **/api/provision_requests**:

```
POST /api/provision_requests
```

```

{
  "version" : "1.1",
  "template_fields" : {
    "guid" : "afe6e8a0-89fd-11e3-b6ac-b8e85646e742"
  },
  "vm_fields" : {
    "number_of_cpus" : 1,
    "vm_name" : "aab_rest_vm1",
    "vm_memory" : "1024",
    "vlan" : "rhevm"
  },
  "requester" : {
    "user_name" : "jdoe",
    "owner_first_name" : "John",
    "owner_last_name" : "Doe",
    "owner_email" : "jdoe@sample.com",
    "auto_approve" : true
  },
  "tags" : {
    "network_location" : "Internal",
    "cc" : "001"
  },
  "additional_values" : {
    "request_id" : "1001"
  },
}

```

```

    "ems_custom_attributes" : { },
    "miq_custom_attributes" : { }
  }

```

Optionally, the action-based request format is also supported:

3.5.1.2. Request:

```
POST /api/provision_requests
```

```

{
  "action" : "create",
  "resource" : {
    "version" : "1.1",
    "template_fields" : {
      "guid" : "afe6e8a0-89fd-11e3-b6ac-b8e85646e742"
    },
    "vm_fields" : {
      "number_of_cpus" : 1,
      "vm_name" : "aab_rest_vm1",
      "vm_memory" : "1024",
      "vlan" : "rhevm"
    },
    "requester" : {
      "user_name" : "jdoe",
      "owner_first_name" : "John",
      "owner_last_name" : "Doe",
      "owner_email" : "jdoe@sample.com",
      "auto_approve" : true
    },
    "tags" : {
      "network_location" : "Internal",
      "cc" : "001"
    },
    "additional_values" : {
      "request_id" : "1001"
    },
    "ems_custom_attributes" : { },
    "miq_custom_attributes" : { }
  }
}

```

3.5.1.3. Response:

```

{
  "results": [
    {
      "id": 18,
      "description": "Provision from [bd-clone-template] to
[aab_rest_vm1]",
      "approval_state": "pending_approval",
      "type": "MiqProvisionRequest",
      "created_on": "2015-05-08T17:42:55Z",
      "updated_on": "2015-05-08T17:42:55Z",

```

```

"requester_id": 1,
"requester_name": "Administrator",
"request_type": "template",
"request_state": "pending",
"message": "VM Provisioning - Request Created",
"status": "Ok",
"options": {
  "use_pre_dialog": false,
  "request_type": "template",
  "miq_request_dialog_name":
"miq_provision_redhat_dialogs_template",
  "owner_first_name": "John",
  "owner_last_name": "Doe",
  "owner_email": "jdoe@sample.com",
  "vm_tags": [
    62,
    58
  ],
  "addr_mode": [
    "static",
    "Static"
  ],
  "placement_cluster_name": [
    null,
    null
  ],
  "cluster_filter": [
    null,
    null
  ],
  "placement_auto": [
    true,
    1
  ],
  "placement_dc_name": [
    null,
    null
  ],
  "number_of_vms": [
    1,
    "1"
  ],
  "src_vm_id": [
    1947,
    "bd-clone-template"
  ],
  "provision_type": [
    "native_clone",
    "Native Clone"
  ],
  "linked_clone": [
    null,
    null
  ],
  "vm_name": "aab_rest_vm1",
  "pxe_server_id": [

```

```
    null,
    null
  ],
  "schedule_type": [
    "immediately",
    "Immediately on Approval"
  ],
  "vm_auto_start": [
    true,
    1
  ],
  "schedule_time": "2015-05-09T13:42:54-04:00",
  "retirement": [
    0,
    "Indefinite"
  ],
  "retirement_warn": [
    604800,
    "1 Week"
  ],
  "stateless": [
    false,
    0
  ],
  "vlan": [
    "rhevm",
    "rhevm"
  ],
  "disk_format": [
    "default",
    "Default"
  ],
  "number_of_sockets": [
    1,
    "1"
  ],
  "cores_per_socket": [
    1,
    "1"
  ],
  "vm_memory": [
    "1024",
    "1024"
  ],
  "network_adapters": [
    1,
    "1"
  ],
  "placement_host_name": [
    null,
    null
  ],
  "placement_ds_name": [
    null,
    null
  ],
  ],
```

```

        "src_vm_nics": [
    ],
    "src_vm_lans": [
    ],
    "customize_enabled": [
        "enabled"
    ],
    "src_ems_id": [
        105,
        "rhevm230"
    ],
    "auto_approve": false,
    "ws_values": {
        "request_id": "1001"
    },
    "ws_ems_custom_attributes": {
    },
    "ws_miq_custom_attributes": {
    }
    },
    "userid": "jdoe",
    "source_id": 1947,
    "source_type": "VmOrTemplate"
}
]
}

```

3.5.2. Triggering Multiple Provision Requests

With provisioning requests:

- **version** defaults to **1.1** if not specified.
- **user_name** defaults to the **REST API** authenticated user if not specified.

3.5.2.1. Request:

POST /api/provision_requests

```

{
  "action" : "create",
  "resources" : [
    {
      "version" : "1.1",
      "template_fields" : { "guid" : "afe6e8a0-89fd-11e3-b6ac-
b8e85646e742" },
      "vm_fields" : {
        "vm_name" : "jdoe_rest_vm1",
        "number_of_cpus" : 1,
        "vm_memory" : "1024",
        "vlan" : "nic1"
      },
      "requester" : {

```

```

    "user_name" : "jdoe",
    "owner_first_name" : "John",
    "owner_last_name" : "Doe",
    "owner_email" : "jdoe@sample.com",
    "auto_approve" : true
  },
  "tags" : {
    "network_location" : "Internal",
    "cc" : "001"
  },
  "additional_values" : { "request_id" : "1001" },
  "ems_custom_attributes" : { },
  "miq_custom_attributes" : { }
},
{
  "template_fields" : { "guid" : "afe6e8a0-89fd-11e3-b6ac-
b8e85646e742" },
  "vm_fields" : {
    "vm_name" : "jdoe_rest_vm2",
    "number_of_cpus" : 1,
    "vm_memory" : "2048",
    "vlan" : "nic1"
  },
  "requester" : {
    "owner_first_name" : "John",
    "owner_last_name" : "Doe",
    "owner_email" : "jdoe@sample.com",
    "auto_approve" : true
  },
  "tags" : {
    "network_location" : "Internal",
    "cc" : "001"
  },
  "additional_values" : { "request_id" : "1002" }
},
{
  "template_fields" : { "guid" : "afe6e8a0-89fd-11e3-b6ac-
b8e85646e742" },
  "vm_fields" : {
    "vm_name" : "jdoe_rest_vm3",
    "number_of_cpus" : 1,
    "vm_memory" : "4096",
    "vlan" : "nic1"
  },
  "requester" : {
    "owner_first_name" : "John",
    "owner_last_name" : "Doe",
    "owner_email" : "jdoe@sample.com",
    "auto_approve" : true
  },
  "tags" : {
    "network_location" : "Internal",
    "cc" : "001"
  },
  "additional_values" : { "request_id" : "1003" }
}

```

```

    }
  ]
}

```

3.5.2.2. Response:

```

{
  "results": [
    {
      "id": 19,
      "description": "Provision from [bd-clone-template] to
[aab_rest_vm1]",
      "approval_state": "pending_approval",
      "type": "MiqProvisionRequest",
      "created_on": "2015-05-08T18:25:25Z",
      "updated_on": "2015-05-08T18:25:26Z",
      "requester_id": 1,
      "requester_name": "jdoe",
      "request_type": "template",
      "request_state": "pending",
      "message": "VM Provisioning - Request Created",
      "status": "Ok",
      "options": {
        "use_pre_dialog": false,
        "request_type": "template",
        "miq_request_dialog_name":
"miq_provision_redhat_dialogs_template",
        "owner_first_name": "John",
        "owner_last_name": "Doe",
        "owner_email": "jdoe@sample.com",
        "vm_tags": [
          62,
          58
        ],
        "addr_mode": [
          "static",
          "Static"
        ],
        "placement_cluster_name": [
          null,
          null
        ],
        "cluster_filter": [
          null,
          null
        ],
        "placement_auto": [
          true,
          1
        ],
        "placement_dc_name": [
          null,
          null
        ],
        "number_of_vms": [

```

```
    1,  
    "1"  
  ],  
  "src_vm_id": [  
    1947,  
    "bd-clone-template"  
  ],  
  "provision_type": [  
    "native_clone",  
    "Native Clone"  
  ],  
  "linked_clone": [  
    null,  
    null  
  ],  
  "vm_name": "aab_rest_vm1",  
  "pxe_server_id": [  
    null,  
    null  
  ],  
  "schedule_type": [  
    "immediately",  
    "Immediately on Approval"  
  ],  
  "vm_auto_start": [  
    true,  
    1  
  ],  
  "schedule_time": "2015-05-09T14:25:25-04:00",  
  "retirement": [  
    0,  
    "Indefinite"  
  ],  
  "retirement_warn": [  
    604800,  
    "1 Week"  
  ],  
  "stateless": [  
    false,  
    0  
  ],  
  "vlan": [  
    "rhevm",  
    "rhevm"  
  ],  
  "disk_format": [  
    "default",  
    "Default"  
  ],  
  "number_of_sockets": [  
    1,  
    "1"  
  ],  
  "cores_per_socket": [  
    1,  
    "1"  
  ]  
]
```



```

    ],
    "vm_memory": [
        "1024",
        "1024"
    ],
    "network_adapters": [
        1,
        "1"
    ],
    "placement_host_name": [
        null,
        null
    ],
    "placement_ds_name": [
        null,
        null
    ],
    "src_vm_nics": [

    ],
    "src_vm_lans": [

    ],
    "customize_enabled": [
        "enabled"
    ],
    "src_ems_id": [
        105,
        "rhevm230"
    ],
    "auto_approve": false,
    "ws_values": {
        "request_id": "1001"
    },
    "ws_ems_custom_attributes": {
    },
    "ws_miq_custom_attributes": {
    }
  },
  "userid": "jdoe",
  "source_id": 1947,
  "source_type": "VmOrTemplate"
},
{
  "id": 20,
  "description": "Provision from [bd-clone-template] to
[aab_rest_vm2]",
  "approval_state": "pending_approval",
  "type": "MiqProvisionRequest",
  "created_on": "2015-05-08T18:25:28Z",
  "updated_on": "2015-05-08T18:25:29Z",
  "requester_id": 1,
  "requester_name": "jdoe",
  "request_type": "template",
  "request_state": "pending",
  "message": "VM Provisioning - Request Created",

```

```
"status": "Ok",
"options": {
  "use_pre_dialog": false,
  "request_type": "template",
  "miq_request_dialog_name":
"miq_provision_redhat_dialogs_template",
  "owner_first_name": "John",
  "owner_last_name": "Doe",
  "owner_email": "jdoe@sample.com",
  "vm_tags": [
    62,
    58
  ],
  "addr_mode": [
    "static",
    "Static"
  ],
  "placement_cluster_name": [
    null,
    null
  ],
  "cluster_filter": [
    null,
    null
  ],
  "placement_auto": [
    true,
    1
  ],
  "placement_dc_name": [
    null,
    null
  ],
  "number_of_vms": [
    1,
    "1"
  ],
  "src_vm_id": [
    1947,
    "bd-clone-template"
  ],
  "provision_type": [
    "native_clone",
    "Native Clone"
  ],
  "linked_clone": [
    null,
    null
  ],
  "vm_name": "aab_rest_vm2",
  "pxe_server_id": [
    null,
    null
  ],
  "schedule_type": [
    "immediately",
```

```
    "Immediately on Approval"
  ],
  "vm_auto_start": [
    true,
    1
  ],
  "schedule_time": "2015-05-09T14:25:28-04:00",
  "retirement": [
    0,
    "Indefinite"
  ],
  "retirement_warn": [
    604800,
    "1 Week"
  ],
  "stateless": [
    false,
    0
  ],
  "vlan": [
    "rhevm",
    "rhevm"
  ],
  "disk_format": [
    "default",
    "Default"
  ],
  "number_of_sockets": [
    1,
    "1"
  ],
  "cores_per_socket": [
    1,
    "1"
  ],
  "vm_memory": [
    "1024",
    "1024"
  ],
  "network_adapters": [
    1,
    "1"
  ],
  "placement_host_name": [
    null,
    null
  ],
  "placement_ds_name": [
    null,
    null
  ],
  "src_vm_nics": [
  ],
  "src_vm_lans": [
```

```

    ],
    "customize_enabled": [
      "enabled"
    ],
    "src_ems_id": [
      105,
      "rhev230"
    ],
    "auto_approve": false,
    "ws_values": {
      "request_id": "1001"
    },
    "ws_ems_custom_attributes": {
    },
    "ws_miq_custom_attributes": {
    }
  },
  "userid": "jdoe",
  "source_id": 1947,
  "source_type": "VmOrTemplate"
},
{
  "id": 21,
  "description": "Provision from [bd-clone-template] to
[aab_rest_vm3]",
  "approval_state": "pending_approval",
  "type": "MiqProvisionRequest",
  "created_on": "2015-05-08T18:25:32Z",
  "updated_on": "2015-05-08T18:25:32Z",
  "requester_id": 1,
  "requester_name": "jdoe",
  "request_type": "template",
  "request_state": "pending",
  "message": "VM Provisioning - Request Created",
  "status": "Ok",
  "options": {
    "use_pre_dialog": false,
    "request_type": "template",
    "miq_request_dialog_name":
"miq_provision_redhat_dialogs_template",
    "owner_first_name": "John",
    "owner_last_name": "Doe",
    "owner_email": "jdoe@sample.com",
    "vm_tags": [
      62,
      58
    ],
  },
  "addr_mode": [
    "static",
    "Static"
  ],
  "placement_cluster_name": [
    null,
    null
  ],
  "cluster_filter": [

```

```

    null,
    null
  ],
  "placement_auto": [
    true,
    1
  ],
  "placement_dc_name": [
    null,
    null
  ],
  "number_of_vms": [
    1,
    "1"
  ],
  "src_vm_id": [
    1947,
    "bd-clone-template"
  ],
  "provision_type": [
    "native_clone",
    "Native Clone"
  ],
  "linked_clone": [
    null,
    null
  ],
  "vm_name": "aab_rest_vm3",
  "pxe_server_id": [
    null,
    null
  ],
  "schedule_type": [
    "immediately",
    "Immediately on Approval"
  ],
  "vm_auto_start": [
    true,
    1
  ],
  "schedule_time": "2015-05-09T14:25:31-04:00",
  "retirement": [
    0,
    "Indefinite"
  ],
  "retirement_warn": [
    604800,
    "1 Week"
  ],
  "stateless": [
    false,
    0
  ],
  "vlan": [
    "rhevm",
    "rhevm"
  ]

```

```
],
  "disk_format": [
    "default",
    "Default"
  ],
  "number_of_sockets": [
    1,
    "1"
  ],
  "cores_per_socket": [
    1,
    "1"
  ],
  "vm_memory": [
    "1024",
    "1024"
  ],
  "network_adapters": [
    1,
    "1"
  ],
  "placement_host_name": [
    null,
    null
  ],
  "placement_ds_name": [
    null,
    null
  ],
  "src_vm_nics": [

  ],
  "src_vm_lans": [

  ],
  "customize_enabled": [
    "enabled"
  ],
  "src_ems_id": [
    105,
    "rhevm230"
  ],
  "auto_approve": false,
  "ws_values": {
    "request_id": "1001"
  },
  "ws_ems_custom_attributes": {
  },
  "ws_miq_custom_attributes": {
  }
},
"userid": "jdoe",
"source_id": 1947,
"source_type": "VmOrTemplate"
```

```

    }
  ]
}

```

3.5.3. Monitoring Request

Once a provisioning request is created, the response result will include the queryable provision request itself, for example: `/api/provision_requests/:id`

3.5.3.1. Response:

```

{
  "results": [
    {
      "id": 3068,
      "description": "Provision from [template1] to [###]",
      "approval_state": "pending_approval",
      "type": "MiqProvisionRequest",
      "created_on": "2015-04-14T17:36:30Z",
      "updated_on": "2015-04-14T17:36:30Z",
      "requester_id": 88913,
      "requester_name": "API User",
      "request_type": "template",
      "request_state": "pending",
      "message": "VM Provisioning - Request Created",
      "status": "Ok"
      "options": {
        "use_pre_dialog": false,
        "request_type": "template",
        "miq_request_dialog_name": "miq_provision_dialogs",
        "src_vm_id": [
          109996,
          "template1"
        ],
        "src_vm_nics": [],
        "src_vm_lans": [],
        "src_ems_id": [
          59136,
          "ems_00000000000002"
        ],
        "placement_auto": [
          true,
          1
        ],
        "vm_tags": [],
        "ws_values": {
        },
        "ws_ems_custom_attributes": {
        },
        "ws_miq_custom_attributes": {
        },
        "tags": {
        }
      },
      "userid": "admin",
    }
  ]
}

```

```

    "source_id": 109996,
    "source_type": "VmOrTemplate"
  }
]
}

```

In the above example, the request can be queried periodically until the `request_state` reaches the `finished` state with:

```
GET /api/provision_requests/3068
```



NOTE

The `requests` tasks of a provisioning request can also be queried by expanding the `request_tasks` subcollection as follows:

```
GET /api/provision_requests/:id?expand=request_tasks
```

An alias `tasks` is also defined for the above subcollection:

```
GET /api/provision_requests/:id?expand=tasks
```

For a list of available attributes, see the [Provisioning Request Attributes](#) section.

3.6. PROVIDERS

This section provides examples of how to interact with providers.

3.6.1. Creating a Provider

3.6.1.1. Request:

```
POST /api/providers
```

```

{
  "type"      : "ManageIQ::Providers::Redhat::InfraManager",
  "name"      : "rhevm101",
  "hostname"  : "rhevm101",
  "ipaddress" : "100.200.300.101",
  "credentials" : {
    "userid"   : "admin_account",
    "password" : "admin_password"
  }
}

```

3.6.1.2. Response:

```

{
  "results": [
    {

```



```

        "id": 105,
        "name": "rhevm101",
        "hostname": "rhevm101",
        "ipaddress": "100.200.300.101",
        "created_on": "2015-05-05T15:47:41Z",
        "updated_on": "2015-05-05T15:47:41Z",
        "guid": "10360312-f33e-11e4-86c7-b8e85646e742",
        "zone_id": 1,
        "type": "ManageIQ::Providers::Redhat::InfraManager"
    }
  ]
}

```

3.6.2. Creating a Provider with Compound Credentials

3.6.2.1. Request:

```
POST /api/providers
```

```

{
  "type"      : "ManageIQ::Providers::Redhat::InfraManager",
  "name"      : "rhevm102",
  "hostname"  : "rhevm102",
  "ipaddress" : "100.200.300.102",
  "credentials" : [
    {
      "userid"   : "default_userid",
      "password" : "default_password"
    },
    {
      "userid"   : "metrics_userid",
      "password" : "metrics_password",
      "auth_type" : "metrics"
    }
  ]
}

```

3.6.2.2. Response:

```

{
  "results": [
    {
      "id": 106,
      "name": "rhevm102",
      "hostname": "rhevm102",
      "ipaddress": "100.200.300.102",
      "created_on": "2015-05-05T15:57:44Z",
      "updated_on": "2015-05-05T15:57:44Z",
      "guid": "acbd610e-f3f6-11e4-aaba-b8e85646e742",
      "zone_id": 1,
      "type": "ManageIQ::Providers::Redhat::InfraManager"
    }
  ]
}

```

```
}  
  ]  
}
```

3.6.3. Refreshing a Provider

3.6.3.1. Request:

```
POST /api/providers/105
```

```
{  
  "action" : "refresh"  
}
```

3.6.3.2. Response:

```
{  
  "success": true,  
  "message": "Provider id:105 name:'rhevm105' refreshing",  
  "href": "https://hostname/api/providers/105"  
}
```

3.6.4. Updating a Provider

3.6.4.1. Request:

```
POST /api/providers/106
```

```
{  
  "action" : "edit",  
  "ipaddress" : "100.200.300.112",  
  "credentials" : [  
    {  
      "userid" : "updated_metrics_userid",  
      "password" : "updated_metrics_password",  
      "auth_type" : "metrics"  
    }  
  ]  
}
```

3.6.4.2. Response:

```
{  
  "href": "https://hostname/api/providers/106",  
  "id": 106,  
  "name": "rhevm102",  
  "hostname": "rhevm102",  
  "ipaddress": "100.200.300.112",  
  "created_on": "2015-05-06T13:49:11Z",  
  "updated_on": "2015-05-06T13:53:06Z",  
}
```

```

    "guid": "acbd610e-f3f6-11e4-aaba-b8e85646e742",
    "zone_id": 1,
    "type": "ManageIQ::Providers::Redhat::InfraManager"
  }

```

3.6.5. Deleting a Provider

3.6.5.1. Deleting a Single Provider

3.6.6. Request:

```
DELETE /api/provider/105
```

Or via the delete action as follows:

```
POST /api/provider/105
```

```

{
  "action" : "delete"
}

```

3.6.7. Response:

```

{
  "success": true,
  "message": "Provider id:106 name:'rhev102' deleting",
  "task_id": 625,
  "task_href": "https://hostname/api/tasks/625",
  "href": "https://hostname/api/providers/106"
}

```



NOTE

Delete actions are done asynchronously as it can take a while to complete. The delete task can be queried as follows:

3.6.8. Request:

```
GET /api/tasks/625
```

3.6.9. Response:

```

{
  "href": "https://hostname/api/tasks/625",
  "id": 625,
  "name": "Provider id:106 name:'rhev102' deleting",
  "state": "Finished",
  "status": "Ok",
  "message": "Task completed successfully",
  "userid": "admin",
}

```

```
"created_on": "2015-05-06T14:02:26Z",  
"updated_on": "2015-05-06T14:02:32Z"  
}
```

3.6.9.1. Deleting Multiple Providers

3.6.10. Request:

```
POST /api/providers
```

```
{  
  "action" : "delete",  
  "resources" : [  
    { "href" : "https://hostname/api/providers/107" },  
    { "href" : "https://hostname/api/providers/108" }  
  ]  
}
```

3.6.11. Response:

```
{  
  "results": [  
    {  
      "success": true,  
      "message": "Provider id:107 name:'rhev102' deleting",  
      "task_id": 626,  
      "task_href": "https://hostname/api/tasks/626",  
      "href": "https://hostname/api/providers/107"  
    },  
    {  
      "success": true,  
      "message": "Provider id:108 name:'rhev103' deleting",  
      "task_id": 627,  
      "task_href": "https://hostname/api/tasks/627",  
      "href": "https://hostname/api/providers/108"  
    }  
  ]  
}
```

3.7. SERVICES

This section provides examples of how to interact with services.

3.7.1. Editing a Service

3.7.1.1. Request:

```
POST /api/services/101
```

```
{  
  "action" : "edit",
```

```

"resource" : {
  "name" : "service_101",
  "description" : "This is an updated description for service 101"
}
}

```

3.7.1.2. Response:

```

{
  "href": "https://hostname/api/services/101",
  "id": 101,
  "name": "service_101",
  "description": "This is an updated description for the service 101",
  "guid": "4a8f96de-a1a6-11e4-9f8d-b8e85646e742",
  "options": {
  },
  "created_at": "2015-01-21T19:47:11Z",
  "updated_at": "2015-04-16T22:17:15Z"
}

```

3.7.2. Editing Multiple Services

3.7.2.1. Request:

```
POST /api/services
```

```

{
  "action" : "edit",
  "resources" : [
    {
      "href" : "https://hostname/api/services/81",
      "description" : "This is an updated description for service 81"
    },
    {
      "href" : "https://hostname/api/services/82",
      "description" : "This is an updated description for service 82"
    }
  ]
}

```

3.7.2.2. Response:

```

{
  "results": [
    {
      "id": 81,
      "name": "api_gen_A_81",
      "description": "This is an updated description for service 81",
      "guid": "eb6daaf8-7c9b-11e4-8a3a-b8e85646e742",
      "options": {
      },
      "created_at": "2014-12-05T16:29:43Z",

```

```

      "updated_at": "2015-04-16T22:43:37Z"
    },
    {
      "id": 82,
      "name": "api_gen_A_82",
      "description": "This is an updated description for service 82",
      "guid": "eb6de400-7c9b-11e4-8a3a-b8e85646e742",
      "options": {
      },
      "created_at": "2014-12-05T16:29:43Z",
      "updated_at": "2015-04-16T22:43:37Z"
    }
  ]
}

```

3.7.3. Editing a Resource with the PATCH Method

Supported attribute actions for PATCH include add, edit and remove.

3.7.3.1. Request:

```
PATCH /api/services/90
```

```

[
  { "action" : "edit", "path" : "name", "value" : "updated_service_90" },
  { "action" : "remove", "path" : "description"}
]

```

3.7.3.2. Response:

```

{
  "href": "https://hostname/api/services/90",
  "id": 90,
  "name": "updated_service_90",
  "guid": "eb6fc61c-7c9b-11e4-8a3a-b8e85646e742",
  "options": {
  },
  "created_at": "2014-12-05T16:29:43Z",
  "updated_at": "2015-04-16T22:38:57Z"
}

```



NOTE

Note that the description attribute is no longer defined for this service.

3.7.4. Editing a Service with the PUT Method

3.7.4.1. Request:

```
PUT /api/services/90
```

```
{
  "name" : "new_service_90",
  "description" : "This is a new description for service 90"
}
```

3.7.4.2. Response:

```
{
  "href": "https://hostname/api/services/90",
  "id": 90,
  "name": "new_service_90",
  "description": "This is a new description for service 90",
  "guid": "eb6fc61c-7c9b-11e4-8a3a-b8e85646e742",
  "options": {
  },
  "created_at": "2014-12-05T16:29:43Z",
  "updated_at": "2015-04-16T22:41:37Z"
}
```

3.7.5. Retiring a Service Immediately

3.7.5.1. Request:

```
POST /api/services
```

```
{
  "action" : "retire",
  "resource" : { "href" : "https://hostname/api/services/35" }
}
```

3.7.5.2. Response:

```
{
  "results" : [
    {
      "href": "https://hostname/api/services/95",
      "id": 95,
      "name": "sample_service_95",
      "description": "Description for sample_service_95",
      "guid": "eb713538-7c9b-11e4-8a3a-b8e85646e742",
      "options": {
      },
      "created_at": "2014-12-05T16:29:43Z",
      "updated_at": "2015-05-08T19:32:00Z"
    }
  ]
}
```

Alternatively, this can be done by targeting the service directly:

```
POST /api/services/95
```

```
{
  "action" : "retire"
}
```

3.7.5.3. Response:

```
{
  "href": "https://hostname/api/services/95",
  "id": 95,
  "name": "sample_service_95",
  "description": "Description for sample_service_95",
  "guid": "eb713538-7c9b-11e4-8a3a-b8e85646e742",
  "options": {
  },
  "created_at": "2014-12-05T16:29:43Z",
  "updated_at": "2015-05-08T19:32:00Z"
}
```

3.7.6. Retiring a Service at a Future Time

3.7.6.1. Request:

```
POST /api/services/93
```

```
{
  "action" : "retire",
  "resource" : { "date" : "10/31/2015", "warn" : "7" }
}
```

3.7.6.2. Response:

```
{
  "href": "https://hostname/api/services/93",
  "id": 93,
  "name": "sample_service_93",
  "description": "Description for sample_service_93",
  "guid": "eb709e02-7c9b-11e4-8a3a-b8e85646e742",
  "options": {
  },
  "created_at": "2014-12-05T16:29:43Z",
  "updated_at": "2015-05-08T19:40:19Z",
  "retired": false,
  "retires_on": "2015-10-30",
  "retirement_warn": 7
}
```

3.7.7. Retiring Multiple Services

3.7.7.1. Request:


```
POST /api/services
```

```
{
  "action" : "retire",
  "resources" : [
    { "href" : "https://hostname/api/services/100" },
    { "href" : "https://hostname/api/services/101", "date" :
"11/01/2015", "warn" : "4" },
    { "href" : "https://hostname/api/services/102", "date" :
"11/02/2015", "warn" : "4" }
  ]
}
```

3.7.7.2. Response:

```
{
  "results": [
    {
      "id": 100,
      "name": "cloud_service_100",
      "description": "Description for cloud_service_100",
      "guid": "eb725f94-7c9b-11e4-8a3a-b8e85646e742",
      "options": {
      },
      "created_at": "2014-12-05T16:29:43Z",
      "updated_at": "2015-05-08T19:46:11Z"
    },
    {
      "id": 101,
      "name": "cloud_service_101",
      "description": "Description for cloud_service_101",
      "guid": "4a8f96de-a1a6-11e4-9f8d-b8e85646e742",
      "options": {
      },
      "created_at": "2015-01-21T19:47:11Z",
      "updated_at": "2015-05-08T19:46:19Z",
      "retired": false,
      "retires_on": "2015-11-01",
      "retirement_warn": 4
    },
    {
      "id": 102,
      "name": "cloud_service_102",
      "description": "Description for cloud_service_102",
      "guid": "e2da0cb0-e47e-11e4-a47f-b8e85646e742",
      "options": {
      },
      "created_at": "2015-04-16T21:23:55Z",
      "updated_at": "2015-05-08T19:46:19Z",
      "retired": false,
      "retires_on": "2015-11-02",
      "retirement_warn": 3
    }
  ]
}
```

```
    }  
  ]  
}
```

3.7.8. Deleting Services

3.7.8.1. Request:

```
POST /api/services
```

```
{  
  "action" : "delete",  
  "resources" : [  
    { "href" : "https://hostname/api/services/97" },  
    { "href" : "https://hostname/api/services/98" },  
    { "href" : "https://hostname/api/services/99" }  
  ]  
}
```

3.7.8.2. Response:

```
{  
  "results": [  
    {  
      "success": true,  
      "message": "services id: 97 deleting",  
      "href": "https://hostname/api/services/97"  
    },  
    {  
      "success": true,  
      "message": "services id: 98 deleting",  
      "href": "https://hostname/api/services/98"  
    },  
    {  
      "success": true,  
      "message": "services id: 99 deleting",  
      "href": "https://hostname/api/services/99"  
    }  
  ]  
}
```

3.8. VIRTUAL MACHINES

This section provides examples of how to interact with virtual machines.

3.8.1. Scanning a Virtual Machine

3.8.1.1. Request:

```
POST /api/vms/1922
```

```
{
  "action": "scan"
}
```

3.8.1.2. Response:

```
{
  "success": true,
  "message": "VM id:1922 name:'aab_test_vm' scanning",
  "task_id": 618,
  "task_href": "https://hostname/api/tasks/618",
  "href": "https://hostname/api/vms/1922"
}
```

Optionally, to query the status of the scan:

3.8.1.3. Request:

```
GET /api/tasks/618
```

3.8.1.4. Response:

```
{
  "href": "https://hostname/api/tasks/618",
  "id": 618,
  "name": "VM id:1922 name:'aab_test_vm' scanning",
  "state": "Finished",
  "status": "Ok",
  "message": "Task completed successfully",
  "userid": "admin",
  "created_on": "2015-05-05T19:37:32Z",
  "updated_on": "2015-05-05T19:37:38Z"
}
```

3.8.2. Setting the Virtual Machine Owner

3.8.2.1. Request:

```
POST /api/vms/1921
```

```
{
  "action": "set_owner",
  "resource" : {
    "owner" : "admin"
  }
}
```

3.8.2.2. Response:

```
{
  "success": true,
```

```
"message": "VM id:1921 name:'aab_demo_vm' setting owner to 'admin'",  
"href": "https://hostname/api/vms/1921"  
}
```

3.8.3. Adding an Event to a Virtual Machine

3.8.3.1. Request:

```
POST /api/vms/1921
```

```
{  
  "action": "add_event",  
  "resource" : {  
    "event_type" : "BadUserNameSessionEvent",  
    "event_message" : "Cannot login user@test.domain"  
  }  
}
```

3.8.3.2. Response:

```
{  
  "success": true,  
  "message": "Adding Event type=BadUserNameSessionEvent message=Cannot  
login user@test.domain",  
  "href": "https://hostname/api/vms/1921"  
}
```

3.8.4. Adding a Lifecycle Event to a Virtual Machine

3.8.4.1. Request:

```
POST /api/vms/1921
```

```
{  
  "action" : "add_lifecycle_event",  
  "resource" : {  
    "event" : "event_name",  
    "status" : "event_status",  
    "message" : "Message about the event",  
    "created_by" : "user_name"  
  }  
}
```

3.8.4.2. Response:

```
{  
  "success": true,  
  "message": "VM id:1921 name:'aab_demo_vm' adding lifecycle
```

```
event=event_name message=Message about the event",
  "href": "https://hostname/api/vms/1921"
}
```

3.8.5. Starting a Virtual Machine

3.8.5.1. Request:

```
POST /api/vms/1921
```

```
{
  "action": "start"
}
```

3.8.5.2. Response:

```
{
  "success": true,
  "message": "VM id:1921 name:'aab_demo_vm' starting",
  "task_id": 610,
  "task_href": "https://hostname/api/tasks/610",
  "href": "https://hostname/api/vms/1921"
}
```

3.8.6. Querying Task Progress

3.8.6.1. Request:

```
GET /api/tasks/620
```

3.8.6.2. Response:

```
{
  "href": "https://hostname/api/tasks/610",
  "id": 610,
  "name": "VM id:1921 name:'aab_demo_vm' starting",
  "state": "Queued",
  "status": "Ok",
  "message": "Queued the action: [VM id:1921 name:'aab_demo_vm' starting]
being run for user: [admin]",
  "userid": "admin",
  "created_on": "2015-05-05T15:58:08Z",
  "updated_on": "2015-05-05T15:58:08Z"
}
```

3.8.7. Stopping a Virtual Machine

3.8.7.1. Request:

```
POST /api/vms/1921
```

```
{  
  "action": "stop"  
}
```

3.8.7.2. Response:

```
{  
  "success": true,  
  "message": "VM id:1921 name:'aab_demo_vm' stopping",  
  "task_id": 619,  
  "task_href": "https://hostname/api/tasks/619",  
  "href": "https://hostname/api/vms/1921"  
}
```

3.8.8. Suspending a Virtual Machine

3.8.8.1. Request:

```
POST /api/vms/1921
```

```
{  
  "action": "suspend"  
}
```

3.8.8.2. Response:

```
{  
  "success": true,  
  "message": "VM id:1921 name:'aab_demo_vm' suspending",  
  "task_id": 620,  
  "task_href": "https://hostname/api/tasks/620",  
  "href": "https://hostname/api/vms/1921"  
}
```

3.8.9. Deleting Virtual Machines

3.8.9.1. Deleting a Single Virtual Machine

3.8.10. Request:

```
DELETE /api/vms/334
```

On success, the virtual machine targeted for deletion asynchronously and no response content with an HTTP status code of 204 are returned.

3.8.10.1. Deleting Multiple Virtual Machines

3.8.11. Request:

```
DELETE /api/vms
```

```
{
  "action" : "delete",
  "resources" : [
    { "href" : "https://hostname/api/vms/348" },
    { "href" : "https://hostname/api/vms/349" },
    { "href" : "https://hostname/api/vms/3" }
  ]
}
```

3.8.12. Response:

```
{
  "results": [
    {
      "success": true,
      "message": "VM id:348 name:'aab-temp1' deleting",
      "task_id": 616,
      "task_href": "https://hostname/api/tasks/616",
      "href": "https://hostname/api/vms/348"
    },
    {
      "success": true,
      "message": "VM id:349 name:'aab-temp2' deleting",
      "task_id": 617,
      "task_href": "https://hostname/api/tasks/617",
      "href": "https://hostname/api/vms/349"
    }
  ]
}
```

Optionally, monitor the asynchronous virtual machine deletion by accessing the related task as follows:

3.8.13. Request:

```
GET /api/tasks/616
```

3.8.14. Response:

```
{
  "href": "https://hostname/api/tasks/616",
  "id": 616,
  "name": "VM id:348 name:'aab-temp1' deleting",
  "state": "Finished",
  "status": "Ok",
  "message": "Task completed successfully",
  "userid": "admin",
  "created_on": "2015-05-05T19:33:35Z",
  "updated_on": "2015-05-05T19:33:40Z"
}
```

```
}

```

3.9. SERVICE TEMPLATES

This section provides examples of how to interact with service templates.

3.9.1. Editing a Service Template

3.9.1.1. Request:

```
POST /api/service_templates/2

```

```
{
  "action" : "edit",
  "resource" : {
    "name" : "updated_svc_template_02",
    "description" : "This is an updated description for service template
02"
  }
}
```

3.9.1.2. Response:

```
{
  "href": "https://hostname/api/service_templates/2",
  "id": 2,
  "name": "updated_svc_template_02",
  "description": "This is an updated description for service template 02",
  "guid": "6f7918b4-d6e7-11e4-9837-b8e85646e742",
  "options": {
  },
  "created_at": "2015-03-30T14:17:02Z",
  "updated_at": "2015-04-16T22:30:02Z",
  "service_type": "unknown",
  "service_template_catalog_id": 6
}
```

3.9.2. Editing Multiple Service Templates

3.9.2.1. Request:

```
POST /api/service_templates

```

```
{
  "action" : "edit",
  "resources" : [
    {
      "href" : "https://hostname/api/service_templates/1",
      "description" : "This is an updated description for the first sample
service template"
    },
  ],
}
```



```

    {
      "href" : "https://hostname/api/service_templates/2",
      "description" : "This is an updated description for the second
sample service template"
    }
  ]
}

```

3.9.2.2. Response:

```

{
  "results": [
    {
      "id": 1,
      "name": "template1",
      "description": "This is an updated description for the first sample
service template",
      "guid": "6a6fdf7e-d6e7-11e4-9837-b8e85646e742",
      "options": {
      },
      "created_at": "2015-03-30T14:16:53Z",
      "updated_at": "2015-04-16T22:33:09Z",
      "service_type": "unknown",
      "service_template_catalog_id": 3
    },
    {
      "id": 2,
      "name": "updated_svc_template_02",
      "description": "This is an updated description for the second sample
service template",
      "guid": "6f7918b4-d6e7-11e4-9837-b8e85646e742",
      "options": {
      },
      "created_at": "2015-03-30T14:17:02Z",
      "updated_at": "2015-04-16T22:33:09Z",
      "service_type": "unknown",
      "service_template_catalog_id": 6
    }
  ]
}

```

3.9.3. Deleting Multiple Service Templates

3.9.3.1. Request:

```
POST /api/service_templates
```

```

{
  "action" : "delete",
  "resources" : [
    { "href" : "https://hostname/api/service_templates/4" },

```

```
    { "href" : "https://hostname/api/service_templates/5" }  
  ]  
}
```

3.9.3.2. Response:

```
{  
  "results": [  
    {  
      "success": true,  
      "message": "service_templates id: 4 deleting",  
      "href": "https://hostname/api/service_templates/4"  
    },  
    {  
      "success": true,  
      "message": "service_templates id: 5 deleting",  
      "href": "https://hostname/api/service_templates/5"  
    }  
  ]  
}
```