Red Hat Virtualization 4.2

Version 3 REST API Guide

Using the Red Hat Virtualization Version 3 REST Application Programming Interface

Last Updated: 2018-11-23

Red Hat Virtualization 4.2 Version 3 REST API Guide

Using the Red Hat Virtualization Version 3 REST Application Programming Interface

Red Hat Virtualization Documentation Team
Red Hat Customer Content Services
rhev-docs@redhat.com

Legal Notice

Abstract

Version 4 of the Red Hat Virtualization Manager supports versions 3 and 4 of the Representational State Transfer API. The information and examples in this guide apply to version 3 of the REST API. This guide is not actively maintained, and is provided as a reference only.

Table of Contents

CHAPTER 1. INTRODUCTION **6**
1.1. REPRESENTATIONAL STATE TRANSFER 6
1.2. RED HAT VIRTUALIZATION REST API PREREQUISITES 6

CHAPTER 2. AUTHENTICATION AND SECURITY **8**
2.1. TLS/SSL CERTIFICATION 8
2.2. HTTP AUTHENTICATION 9
2.3. AUTHENTICATION SESSIONS 10

CHAPTER 3. REST API QUICK START EXAMPLE **12**
3.1. EXAMPLE: ACCESS API ENTRY POINT 12
3.2. EXAMPLE: LIST DATA CENTER COLLECTION 14
3.3. EXAMPLE: LIST HOST CLUSTER COLLECTION 16
3.4. EXAMPLE: LIST LOGICAL NETWORKS COLLECTION 17
3.5. EXAMPLE: LIST HOST COLLECTION 18
3.6. EXAMPLE: LIST CPU PROFILES 20
3.7. EXAMPLE: CREATE NFS DATA STORAGE 21
3.8. EXAMPLE: CREATE NFS ISO STORAGE 22
3.9. EXAMPLE: ATTACH STORAGE DOMAINS TO DATA CENTER 24
3.10. EXAMPLE: ACTIVATE STORAGE DOMAINS 25
3.11. EXAMPLE: CREATE VIRTUAL MACHINE 26
3.12. EXAMPLE: CREATE VIRTUAL MACHINE NIC 30
3.13. EXAMPLE: CREATE VIRTUAL MACHINE STORAGE DISK 30
3.14. EXAMPLE: ATTACH ISO IMAGE TO VIRTUAL MACHINE 31
3.15. EXAMPLE: START VIRTUAL MACHINE 32
3.16. EXAMPLE: CHECK SYSTEM EVENTS 33

CHAPTER 4. ENTRY POINT **35**
4.1. PRODUCT INFORMATION 36
4.2. LINK ELEMENTS 36
4.3. SPECIAL OBJECT ELEMENTS 38
4.4. SUMMARY ELEMENT 39
4.5. RESTFUL SERVICE DESCRIPTION LANGUAGE (RSDL) 39
4.6. RED HAT VIRTUALIZATION WINDOWS GUEST VSS SUPPORT 41
4.7. QEMU GUEST AGENT OVERVIEW 41
4.8. VSS TRANSACTION FLOW 42

CHAPTER 5. COMPATIBILITY LEVEL VERSIONS **43**
5.1. UPGRADING COMPATIBILITY LEVELS 43

CHAPTER 6. CAPABILITIES **45**
6.1. VERSION-DEPENDENT CAPABILITIES 45
6.2. CURRENT VERSION 45
6.3. FEATURES 46

CHAPTER 7. COMMON FEATURES **49**
7.1. ELEMENT PROPERTY ICONS 49
7.2. REPRESENTATIONS 49
7.3. COLLECTIONS 50
7.4. RESOURCES 56

CHAPTER 8. THE BACKUP AND RESTORE API **63**
8.1. BACKING UP A VIRTUAL MACHINE 63

8.2. RESTORING A VIRTUAL MACHINE 64

CHAPTER 9. DATA CENTERS 66
9.1. DATA CENTER ELEMENTS 66
9.2. XML REPRESENTATION OF A DATA CENTER 67
9.3. JSON REPRESENTATION OF A DATA CENTER 68
9.4. METHODS 69
9.5. SUB-COLLECTIONS 70
9.6. ACTIONS 75

CHAPTER 10. CLUSTERS 77
10.1. CLUSTER ELEMENTS 77
10.2. MEMORY POLICY ELEMENTS 79
10.3. SCHEDULING POLICY ELEMENTS 79
10.4. XML REPRESENTATION OF A CLUSTER 80
10.5. JSON REPRESENTATION OF A CLUSTER 81
10.6. METHODS 82
10.7. SUB-COLLECTIONS 83

CHAPTER 11. NETWORKS 93
11.1. NETWORK ELEMENTS 93
11.2. XML REPRESENTATION OF A NETWORK RESOURCE 93
11.3. JSON REPRESENTATION OF A NETWORK RESOURCE 94
11.4. METHODS 95
11.5. SUB-COLLECTIONS 96

CHAPTER 12. STORAGE DOMAINS 98
12.1. STORAGE DOMAIN ELEMENTS 98
12.2. XML REPRESENTATION OF A STORAGE DOMAIN 99
12.3. JSON REPRESENTATION OF A STORAGE DOMAIN 100
12.4. METHODS 101
12.5. STORAGE TYPES 102
12.6. EXPORT STORAGE DOMAINS 105
12.7. GLANCE IMAGE STORAGE DOMAINS 107
12.8. IMPORTING A BLOCK STORAGE DOMAIN 109
12.9. SUB-COLLECTIONS 112
12.10. ACTIONS 113

CHAPTER 13. STORAGE CONNECTIONS 116
13.1. STORAGE CONNECTION ELEMENTS 116
13.2. XML REPRESENTATION OF A STORAGE CONNECTION RESOURCE 117
13.3. METHODS 117

CHAPTER 14. HOSTS 121
14.1. HOST ELEMENTS 121
14.2. XML REPRESENTATION OF A HOST 124
14.3. JSON REPRESENTATION OF A HOST 127
14.4. POWER MANAGEMENT ELEMENTS 130
14.5. MEMORY MANAGEMENT ELEMENTS 132
14.6. METHODS 133
14.7. SUB-COLLECTIONS 134
14.8. ACTIONS 153

CHAPTER 15. VIRTUAL MACHINES 158
15.1. VIRTUAL MACHINE ELEMENTS 158

15.2. XML REPRESENTATION OF A VIRTUAL MACHINE 164
15.3. XML REPRESENTATION OF ADDITIONAL OVF DATA FOR A VIRTUAL MACHINE 167
15.4. JSON REPRESENTATION OF A VIRTUAL MACHINE 169
15.5. METHODS 173
15.6. SUB-COLLECTIONS 176
15.7. ACTIONS 203

CHAPTER 16. FLOATING DISKS 212
16.1. FLOATING DISK ELEMENTS 212
16.2. XML REPRESENTATION OF A FLOATING DISK 213
16.3. METHODS 214
16.4. SUB-COLLECTIONS 214
16.5. ACTIONS 216

CHAPTER 17. TEMPLATES 217
17.1. VIRTUAL MACHINE TEMPLATE ELEMENTS 217
17.2. XML REPRESENTATION OF A VIRTUAL MACHINE TEMPLATE 219
17.3. METHODS 221
17.4. ACTIONS 223

CHAPTER 18. VIRTUAL MACHINE POOLS 224
18.1. VIRTUAL MACHINE POOL ELEMENTS 224
18.2. XML REPRESENTATION OF A VIRTUAL MACHINE POOL 224
18.3. METHODS 225
18.4. ACTIONS 226

CHAPTER 19. DOMAINS 227
19.1. DOMAIN ELEMENTS 227
19.2. XML REPRESENTATION OF A DOMAIN RESOURCE 227
19.3. SUB-COLLECTIONS 227

CHAPTER 20. GROUPS 230
20.1. IMPORTED GROUP ELEMENTS 230
20.2. XML REPRESENTATION OF A GROUP RESOURCE 230
20.3. ADDING A GROUP FROM A DIRECTORY SERVICE 230

CHAPTER 21. ROLES 232
21.1. ROLE ELEMENTS 232
21.2. XML REPRESENTATION OF THE ROLES COLLECTION 232
21.3. METHODS 233
21.4. ROLES PERMITS SUB-COLLECTION 234

CHAPTER 22. USERS 236
22.1. USER ELEMENTS 236
22.2. XML REPRESENTATION OF A USER RESOURCE 236
22.3. METHODS 237

CHAPTER 23. MAC ADDRESS POOLS 239
23.1. MAC ADDRESS POOL ELEMENTS 239
23.2. XML REPRESENTATION OF THE MAC ADDRESS POOLS COLLECTION 239
23.3. METHODS 240

CHAPTER 24. TAGS 242
24.1. TAG ELEMENTS 242
24.2. XML REPRESENTATION OF A TAG RESOURCE 242
24.3. ASSOCIATING TAGS 242

24.4. PARENT TAGS 244

CHAPTER 25. EVENTS 247
25.1. EVENT ELEMENTS 247
25.2. XML REPRESENTATION OF THE EVENTS COLLECTION 247
25.3. XML REPRESENTATION OF A VIRTUAL MACHINE CREATION EVENT 248
25.4. METHODS 248

APPENDIX A. API USAGE WITH CURL 252
A.1. API USAGE WITH CURL 252
A.2. INSTALLING CURL 252
A.3. USING CURL 252
A.4. EXAMPLES 253

CHAPTER 1. INTRODUCTION

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

The benefits of the REST API are:

- Broad client support - Any programming language, framework, or system with support for HTTP protocol can use the API;
- Self descriptive - Client applications require minimal knowledge of the virtualization infrastructure as many details are discovered at runtime;
- Resource-based model - The resource-based REST model provides a natural way to manage a virtualization platform.

This provides developers and administrators with the ability to:

- Integrate with enterprise IT systems.
- Integrate with third-party virtualization software.
- Perform automated maintenance or error checking tasks.
- Automate repetitive tasks in a Red Hat Virtualization environment with scripts.

This documentation acts as a reference to the Red Hat Virtualization Manager REST API. It aims to provide developers and administrators with instructions and examples to help harness the functionality of their Red Hat Virtualization environment through the REST API either directly or using the provided Python libraries.

1.1. REPRESENTATIONAL STATE TRANSFER

Representational State Transfer (REST) is a design architecture that focuses on resources for a specific service and their representations. A resource representation is a key abstraction of information that corresponds to one specific managed element on a server. A client sends a request to a server element located at a Uniform Resource Identifier (URI) and performs operations with standard HTTP methods, such as **`GET`**, **`POST`**, **`PUT`**, and **`DELETE`**. This provides a stateless communication between the client and server where each request acts independent of any other request and contains all necessary information to complete the request.

1.2. RED HAT VIRTUALIZATION REST API PREREQUISITES

Red Hat Virtualization REST API Prerequisites

- A networked installation of Red Hat Virtualization Manager, which includes the REST API.
- A client or programming library that initiates and receives HTTP requests from the REST API. For example:
  - Python software development kit (SDK)

    - Java software development kit (SDK)
    - **cURL** command line tool
    - **RESTClient**, a debugger for RESTful web services

- Knowledge of Hypertext Transfer Protocol (HTTP), which is the protocol used for REST API interactions. The Internet Engineering Task Force provides a Request for Comments (RFC) explaining the Hypertext Transfer Protocol at http://www.ietf.org/rfc/rfc2616.txt.
- Knowledge of Extensible Markup Language (XML) or JavaScript Object Notation (JSON), which the API uses to construct resource representations. The W3C provides a full specification on XML at http://www.w3.org/TR/xml/. ECMA International provide a free publication on JSON at http://www.ecma-international.org.

CHAPTER 2. AUTHENTICATION AND SECURITY

2.1. TLS/SSL CERTIFICATION

The Red Hat Virtualization Manager API requires Hypertext Transfer Protocol Secure (HTTPS) [1] for secure interaction with client software, such as the Manager's SDK and CLI components. This involves a process of obtaining a certificate from the Red Hat Virtualization Manager and importing it into the certificate store of your client.

IMPORTANT

Obtain your certificate from the Red Hat Virtualization Manager using a secure network connection.

Procedure 2.1. Obtaining a Certificate

You can obtain a certificate from the Red Hat Virtualization Manager and transfer it to the client machine using one of three methods:

1. **Method 1** - Use a command line tool to download the certificate from the Manager. Examples of command line tools include **cURL** and **Wget**, both of which are available on multiple platforms.

   a. If using **cURL**:

   ```
   $ curl -o rhvm.cer http://[manager-fqdn]/ovirt-
   engine/services/pki-resource?resource=ca-certificate&format=X509-
   PEM-CA
   ```

   b. If using **Wget**:

   ```
   $ wget -O rhvm.cer http://[manager-fqdn]/ovirt-
   engine/services/pki-resource?resource=ca-certificate&format=X509-
   PEM-CA
   ```

2. **Method 2** - Use a web browser to navigate to the certificate located at:

   ```
   http://[manager-fqdn]/ovirt-engine/services/pki-resource?
   resource=ca-certificate&format=X509-PEM-CA
   ```

   Depending on the chosen browser, the certificate either downloads or imports into the browser's keystore.

   a. **If the browser downloads the certificate:** save the file as **`rhvm.cer`**.

   If the browser imports the certificate: export it from the browser's certification options and save it as **`rhvm.cer`**.

3. **Method 3** - Log in to the Manager, export the certificate from the truststore and copy it to your client machine.

   a. Log in to the Manager as the **`root`** user.

   b. Export the certificate from the truststore using the Java **keytool** management utility:

```
$ keytool -exportcert -keystore /etc/pki/ovirt-engine/.truststore
-alias cacert -storepass mypass -file rhvm.cer
```

This creates a certificate file called **rhvm.cer**.

c. Copy the certificate to the client machine using the **scp** command:

```
$ scp rhvm.cer [username]@[client-machine]:[directory]
```

Each of these methods results in a certificate file named **rhvm.cer** on your client machine. An API user imports this file into the certificate store of the client.

Procedure 2.2. Importing a Certificate to a Client

- Importing a certificate to a client relies on how the client itself stores and interprets certificates. This guide contains some examples on importing certificates. For clients not using Network Security Services (NSS) or Java KeyStore (JKS), see your client documentation for more information on importing a certificate.

2.2. HTTP AUTHENTICATION

Any user with a Red Hat Virtualization account has access to the REST API. An API user submits a mandatory Red Hat Virtualization Manager user name and password with all requests to the API. Each request uses HTTP Basic Authentication [2] to encode these credentials. If a request does not include an appropriate **Authorization** header, the API sends a **401 Authorization Required** as a result:

Example 2.1. Access to the REST API without appropriate credentials

```
HEAD [base] HTTP/1.1
Host: [host]

HTTP/1.1 401 Authorization Required
```

Request are issued with an **Authorization** header for the specified realm. An API user encodes an appropriate Red Hat Virtualization Manager domain and user in the supplied credentials with the **username@domain:password** convention.

The following table shows the process for encoding credentials in base64.

Table 2.1. Encoding credentials for API access

Item	Value
username	rhevmadmin
domain	domain.example.com
password	123456
unencoded credentials	rhevmadmin@domain.example.com:123456

Item	Value
base64 encoded credentials	cmhldm1hZG1pbkBibGFjay5xdW1yYW5ldC5jb206MTIzNDU2

An API user provides the base64 encoded credentials as shown:

Example 2.2. Access to the REST API with appropriate credentials

```
HEAD [base] HTTP/1.1
Host: [host]
Authorization: Basic cmhldm1hZG1pbkBibGFjay5xdW1yYW5ldC5jb206MTIzNDU2

HTTP/1.1 200 OK
...
```

IMPORTANT

Basic authentication involves potentially sensitive information, such as passwords, sent as plain text. REST API requires Hypertext Transfer Protocol Secure (HTTPS) for transport-level encryption of plain-text requests.

IMPORTANT

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

2.3. AUTHENTICATION SESSIONS

The API also provides the ability for authentication session support. An API user sends an initial request with authentication details, then sends all subsequent requests using a session cookie to authenticate. The following procedure demonstrates how to use an authenticated session.

Procedure 2.3. Requesting an authenticated session

1. Send a request with the **`Authorization`** and **`Prefer: persistent-auth`**

```
HEAD [base] HTTP/1.1
Host: [host]
Authorization: Basic
cmhldm1hZG1pbkBibGFjay5xdW1yYW5ldC5jb206MTIzNDU2
Prefer: persistent-auth

HTTP/1.1 200 OK
...
```

This returns a response with the following header:

```
Set-Cookie: JSESSIONID=5dQja5ubr4yvI2MM2z+LZxrK; Path=/ovirt-
engine/api; Secure
```

Note the **JSESSIONID=** value. In this example the value is **JSESSIONID=5dQja5ubr4yvI2MM2z+LZxrK**.

2. Send all subsequent requests with the **Prefer: persistent-auth** and **cookie** header with the **JSESSIONID=** value. The **Authorization** is no longer needed when using an authenticated session.

```
HEAD [base] HTTP/1.1
Host: [host]
Prefer: persistent-auth
cookie: JSESSIONID=5dQja5ubr4yvI2MM2z+LZxrK

HTTP/1.1 200 OK
...
```

3. When the session is no longer required, perform a request to the sever without the **Prefer: persistent-auth** header.

```
HEAD [base] HTTP/1.1
Host: [host]
Authorization: Basic
cmhldm1hZG1pbkBibGFjay5xdW1yYW5ldC5jb206MTIzNDU2

HTTP/1.1 200 OK
...
```

[1] HTTPS is described in RFC 2818 HTTP Over TLS.

[2] Basic Authentication is described in RFC 2617 HTTP Authentication: Basic and Digest Access Authentication.

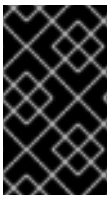
CHAPTER 3. REST API QUICK START EXAMPLE

This chapter provides an example to demonstrate the REST API's ability to setup a basic Red Hat Virtualization environment and create a virtual machine.

In addition to the standard prerequisites, this example requires the following:

- A networked and configured Red Hat Virtualization Host;
- An ISO file containing a desired virtual machine operating system to install. This chapter uses Red Hat Enterprise Linux Server 6 for our installation ISO example; and
- Red Hat Virtualization's **engine-iso-uploader** tool to upload your chosen operating system ISO file.

This example uses **cURL** to demonstrate REST requests with a client application. Note that any application capable of HTTP requests can substitute for **cURL**.



IMPORTANT

For simplicity, the HTTP request headers in this example omit the **`Host:`** and **`Authorization:`** fields. However, these fields are mandatory and require data specific to your installation of Red Hat Virtualization Manager.

IMPORTANT

All **cURL** examples include placeholders for authentication details (**USER:PASS**) and certificate location (**CERT**). Ensure all requests performed with **cURL** fulfill certification and authentication requirements.

NOTE

Red Hat Virtualization Manager generates a globally unique identifier (GUID) for the **`id`** attribute for each resource. Identifier codes in this example might appear different to the identifier codes in your Red Hat Virtualization environment.

3.1. EXAMPLE: ACCESS API ENTRY POINT

The following request retrieves a representation of the main entry point for version 3 of the API.

Example 3.1. Access the API v3 entry point

Request (with header):

```
GET /ovirt-engine/api HTTP/1.1
Version: 3
Accept: application/xml
```

Request (without header):

```
GET /ovirt-engine/api/v3 HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] https://[RHEVM Host]:443/ovirt-engine/api
```

Result:

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

<api>
    <link rel="capabilities" href="/ovirt-engine/api/capabilities"/>
    <link rel="clusters" href="/ovirt-engine/api/clusters"/>
    <link rel="clusters/search" href="/ovirt-engine/api/clusters?search=
{query}"/>
    <link rel="datacenters" href="/ovirt-engine/api/datacenters"/>
    <link rel="datacenters/search" href="/ovirt-engine/api/datacenters?
search={query}"/>
    <link rel="events" href="/ovirt-engine/api/events"/>
    <link rel="events/search" href="/ovirt-engine/api/events?search=
{query}"/>
    <link rel="hosts" href="/ovirt-engine/api/hosts"/>
    <link rel="hosts/search" href="/ovirt-engine/api/hosts?search=
{query}"/>
    <link rel="networks" href="/ovirt-engine/api/networks"/>
    <link rel="roles" href="/ovirt-engine/api/roles"/>
    <link rel="storagedomains" href="/ovirt-engine/api/storagedomains"/>
    <link rel="storagedomains/search" href="/ovirt-
engine/api/storagedomains?search={query}"/>
    <link rel="tags" href="/ovirt-engine/api/tags"/>
    <link rel="templates" href="/ovirt-engine/api/templates"/>
    <link rel="templates/search" href="/ovirt-engine/api/templates?
search={query}"/>
    <link rel="users" href="/ovirt-engine/api/users"/>
    <link rel="groups" href="/ovirt-engine/api/groups"/>
    <link rel="domains" href="/ovirt-engine/api/domains"/>
    <link rel="vmpools" href="/ovirt-engine/api/vmpools"/>
    <link rel="vmpools/search" href="/ovirt-engine/api/vmpools?search=
{query}"/>
    <link rel="vms" href="/ovirt-engine/api/vms"/>
    <link rel="vms/search" href="/ovirt-engine/api/vms?search={query}"/>
    <special_objects>
        <link rel="templates/blank"
          href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000"/>
        <link rel="tags/root"
          href="/ovirt-engine/api/tags/00000000-0000-0000-0000-
000000000000"/>
    </special_objects>
    <product_info>
        <name>Red Hat Virtualization</name>
        <vendor>Red Hat</vendor>
        <version revision="0" build="0" minor="0" major="4"/>
    </product_info>
    <summary>
```

```
        <vms>
            <total>5</total>
            <active>0</active>
        </vms>
        <hosts>
            <total>1</total>
            <active>1</active>
        </hosts>
        <users>
            <total>1</total>
            <active>1</active>
        </users>
        <storage_domains>
            <total>2</total>
            <active>2</active>
        </storage_domains>
    </summary>
</ovirt-engine/api>
```

IMPORTANT

When neither the header nor the URL prefix are used, the server will automatically select a version. The default is version **4**. You can change the default version using the **ENGINE_API_DEFAULT_VERSION** parameter:

```
# echo "ENGINE_API_DEFAULT_VERSION=3" > \
  /etc/ovirt-engine/engine.conf.d/99-set-default-version.conf
# systemctl restart ovirt-engine
```

Changing this parameter affects all users of the Manager that don't specify the version explicitly.

The entry point provides a user with links to the collections in a virtualization environment. The **rel=** attribute of each collection link provides a reference point for each link. The next step in this example examines the **datacenter** collection, which is available through the **rel="datacenter"** link.

The entry point also contains other data such as **product_info**, **special_objects** and **summary**. This data is covered in chapters outside this example.

3.2. EXAMPLE: LIST DATA CENTER COLLECTION

Red Hat Virtualization Manager creates a **Default** data center on installation. This example uses the **Default** data center as the basis for our virtual environment.

The following request retrieves a representation of the data center collection:

Example 3.2. List data center collection

Request:

```
GET /ovirt-engine/api/datacenters HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/datacenters
```

Result:

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

<data_centers>
    <data_center href="/ovirt-engine/api/datacenters/00000002-0002-0002-
0002-0000000003ab" id="00000002-0002-0002-0002-0000000003ab">
        <name>Default</name>
        <description>The default Data Center</description>
        <link rel="storagedomains"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/storagedomains"
        <link rel="clusters"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/clusters"
        <link rel="networks"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/networks"
        <link rel="permissions"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/permissions"
        <link rel="quotas"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/quotas"
        <link rel="iscsibonds"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/iscsibonds"
        <link rel="qoss"/>
          href="/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
0000000003ab/qoss"
        <local>false</local>
        <storage_format>v3</storage_format>
        <version major="4" minor="0"/>
        <supported_versions>
            <version major="4" minor="0"/>
        </supported_versions>
        <status>
            <state>up</state>
        </status>
    </data_center>
</data_centers>
```

Note the **id** code of your **Default** data center. This code identifies this data center in relation to other resources of your virtual environment.

The data center also contains a link to the **storagedomains** sub-collection. The data center uses this sub-collection to attach storage domains from the **storagedomains** main collection, which this example covers later.

3.3. EXAMPLE: LIST HOST CLUSTER COLLECTION

Red Hat Virtualization Manager creates a **Default** host cluster on installation. This example uses the **Default** cluster to group resources in your Red Hat Virtualization environment.

The following request retrieves a representation of the cluster collection:

Example 3.3. List host clusters collection

Request:

```
GET /ovirt-engine/api/clusters HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/clusters
```

Result:

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

<clusters>
    <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"
      href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95">
        <name>Default</name>
        <description>The default server cluster</description>
        <link rel="networks"
          href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks"/>
        <link rel="permissions"
          href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/permissions"/>
        <cpu id="Intel Penryn Family"/>
        <data_center id="01a45ff0-915a-11e0-8b87-5254004ac988"
          href="/ovirt-engine/api/datacenters/01a45ff0-915a-11e0-8b87-
5254004ac988"/>
        <memory_policy>
            <overcommit percent="100"/>
            <transparent_hugepages>
                <enabled>false</enabled>
            </transparent_hugepages>
        </memory_policy>
        <scheduling_policy/>
        <version minor="0" major="4"/>
        <error_handling>
```

```
            <on_error>migrate</on_error>
        </error_handling>
    </cluster>
</clusters>
```

Note the **id** code of your **Default** host cluster. This code identifies this host cluster in relation to other resources of your virtual environment.

The **Default** cluster is associated with the **Default** data center through a relationship using the **id** and **href** attributes of the **data_center** element.

The **networks** sub-collection contains a list of associated network resources for this cluster. The next section examines the **networks** collection in more detail.

3.4. EXAMPLE: LIST LOGICAL NETWORKS COLLECTION

Red Hat Virtualization Manager creates a default **ovirtmgmt** network on installation. This network acts as the management network for Red Hat Virtualization Manager to access hosts.

This network is associated with our **Default** cluster and is a member of the **Default** data center. This example uses the **ovirtmgmt** network to connect our virtual machines.

The following request retrieves a representation of the logical networks collection:

Example 3.4. List logical networks collection

Request:

```
GET /ovirt-engine/api/networks HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/networks
```

Result:

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

<networks>
    <network id="00000000-0000-0000-0000-000000000009"
      href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009">
        <name>ovirtmgmt</name>
        <description>Management Network</description>
        <data_center id="01a45ff0-915a-11e0-8b87-5254004ac988"
          href="/ovirt-engine/api/datacenters/01a45ff0-915a-11e0-8b87-
5254004ac988"/>
        <stp>false</stp>
```

```
        <status>
            <state>operational</state>
        </status>
        <display>false</display>
    </network>
</networks>
```

The **ovirtmgmt** network is attached to the **Default** data center through a relationship using the data center's **id** code.

The **ovirtmgmt** network is also attached to the **Default** cluster through a relationship in the cluster's **network** sub-collection.

3.5. EXAMPLE: LIST HOST COLLECTION

This example uses a Red Hat Virtualization Host. Red Hat Virtualization Manager automatically registers any configured Red Hat Virtualization Host. This example retrieves a representation of the hosts collection and shows a Red Hat Virtualization Host named **hypervisor** registered with the virtualization environment.

Example 3.5. List hosts collection

Request:

```
GET /ovirt-engine/api/hosts HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/hosts
```

Result:

```
HTTP/1.1 200 OK
Accept: application/xml

<hosts>
    <host id="0656f432-923a-11e0-ad20-5254004ac988"
      href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988">
        <name>hypervisor</name>
        <actions>
            <link rel="install"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/install"/>
            <link rel="activate"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/activate"/>
            <link rel="fence"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
```

```
5254004ac988/fence"/>
            <link rel="deactivate"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/deactivate"/>
            <link rel="approve"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/approve"/>
            <link rel="iscsilogin"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/iscsilogin"/>
            <link rel="iscsidiscover"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/iscsidiscover"/>
            <link rel="commitnetconfig"
              href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/
              commitnetconfig"/>
        </actions>
        <link rel="storage"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/storage"/>
        <link rel="nics"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/nics"/>
        <link rel="tags"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/tags"/>
        <link rel="permissions"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/permissions"/>
        <link rel="statistics"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988/statistics"/>
        <address>10.64.14.110</address>
        <status>
            <state>non_operational</state>
        </status>
        <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"
          href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95"/>
        <port>54321</port>
        <storage_manager>true</storage_manager>
        <power_management>
            <enabled>false</enabled>
            <options/>
        </power_management>
        <ksm>
            <enabled>false</enabled>
        </ksm>
        <transparent_hugepages>
            <enabled>true</enabled>
        </transparent_hugepages>
        <iscsi>
            <initiator>iqn.1994-05.com.example:644949fe81ce</initiator>
        </iscsi>
        <cpu>
```

```
            <topology cores="2"/>
            <name>Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00GHz</name>
            <speed>2993</speed>
        </cpu>
        <summary>
            <active>0</active>
            <migrating>0</migrating>
            <total>0</total>
        </summary>
    </host>
</hosts>
```

Note the **id** code of your **Default** host. This code identifies this host in relation to other resources of your virtual environment.

This host is a member of the **Default** cluster and accessing the **nics** sub-collection shows this host has a connection to the **ovirtmgmt** network.

3.6. EXAMPLE: LIST CPU PROFILES

The following request retrieves a representation of the CPU profiles:

Example 3.6. List CPU profiles

Request:

```
GET /ovirt-engine/api/cpuprofiles HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert
[CERT] https://[RHEVM Host]:443/ovirt-engine/api/cpuprofiles
```

Result:

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

<cpu_profiles>
    <cpu_profile href="0000001a-001a-001a-001a-00000000035e"
id="0000001a-001a-001a-001a-00000000035e">
        <name>Default</name>
        <link href="/ovirt-engine/api/cpuprofiles/0000001a-001a-001a-
001a-00000000035e/permissions" rel="permissions"/>
        <cluster href= "/ovirt-engine/api/clusters/00000001-0001-0001-
0001-00000000021b" id="00000001-0001-0001-0001-00000000021b"/>
    </cpu_profile>
    <cpu_profile href="fc4b9188-f87f-44f9-b9c5-c7665e10e0a2"
id="fc4b9188-f87f-44f9-b9c5-c7665e10e0a2">
        <name>Premium</name>
        <description>Full service available</description>
        <link href="/ovirt-engine/api/cpuprofiles/fc4b9188-f87f-44f9-
```

```
b9c5-c7665e10e0a2/permissions" rel="permissions"/>
        <qos href= "/ovirt-engine/api/datacenters/00000002-0002-0002-
0002-0000000000f7/qoss/5afe49e3-aac4-4b7b-bb83-11b9aef285e1"
id="5afe49e3-aac4-4b7b-bb83-11b9aef285e1"/>
        <cluster href= "/ovirt-engine/api/clusters/00000001-0001-0001-
0001-00000000021b" id="00000001-0001-0001-0001-00000000021b"/>
    </cpu_profile>
    <cpu_profile href="48c600f4-6768-49ca-9c16-a877d0e586e5"
id="48c600f4-6768-49ca-9c16-a877d0e586e5">
        <name>Budget</name>
        <description>Limited CPU</description>
        <link href="/ovirt-engine/api/cpuprofiles/48c600f4-6768-49ca-
9c16-a877d0e586e5/permissions" rel="permissions"/>
        <cluster href= "/ovirt-engine/api/clusters/00000001-0001-0001-
0001-00000000021b" id="00000001-0001-0001-0001-00000000021b"/>
    </cpu_profile>
    <cpu_profile href="48c600f4-6768-49ca-9c16-a877d0e586e5"
id="48c600f4-6768-49ca-9c16-a877d0e586e5">
        <name>Backup</name>
        <link href="/ovirt-engine/api/cpuprofiles/d510b042-42f0-4cb2-
9d2e-25fcc28d6c5f/permissions" rel="permissions"/>
        <cluster href= "/ovirt-engine/api/clusters/668cab0c-9185-4eaa-
9942-658284eeecdd" id="668cab0c-9185-4eaa-9942-658284eeecdd"/>
    </cpu_profile>
</cpu_profiles>
```

3.7. EXAMPLE: CREATE NFS DATA STORAGE

An NFS data storage domain is an exported NFS share attached to a data center and provides storage for virtualized guest images. Creation of a new storage domain requires a **POST** request, with the storage domain representation included, sent to the URL of the storage domain collection.

You can enable the wipe after delete option by default on the storage domain. To configure this specify **<wipe_after_delete>** in the **POST** request. This option can be edited after the domain is created, but doing so will not change the wipe after delete property of disks that already exist.

Example 3.7. Create an NFS data storage domain

Request:

```
POST /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
  <name>data1</name>
  <type>data</type>
  <storage>
    <type>nfs</type>
    <address>192.168.0.10</address>
    <path>/data1</path>
  </storage>
  <host>
    <name>hypervisor</name>
```

```
  </host>
</storage_domain>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<storage_domain><name>data1</name><type>data</type> \
    <storage><type>nfs</type><address>192.168.0.10</address> \
    <path>/data1</path></storage> \
    <host><name>hypervisor</name></host></storage_domain>" \
    https://[RHEVM Host]:443/ovirt-engine/api/storagedomains
```

The API creates a NFS data storage domain called **data1** with an export path of **192.168.0.10:/data1** and sets access to the storage domain through the **hypervisor** host. The API also returns the following representation of the newly created storage domain resource.

Result:

```
HTTP/1.1 200 OK
Accept: application/xml

<storage_domain id="9ca7cb40-9a2a-4513-acef-dc254af57aac"
  href="/ovirt-engine/api/storagedomains/9ca7cb40-9a2a-4513-acef-
dc254af57aac">
    <name>data1</name>
    <link rel="permissions"
      href="/ovirt-engine/api/storagedomains/9ca7cb40-9a2a-4513-acef-
dc254af57aac/
      permissions"/>
    <link rel="files"
      href="/ovirt-engine/api/storagedomains/9ca7cb40-9a2a-4513-acef-
dc254af57aac/files"/>
    <type>data</type>
    <master>false</master>
    <storage>
        <type>nfs</type>
        <address>192.168.0.10</address>
        <path>/data1</path>
    </storage>
    <available>175019917312</available>
    <used>27917287424</used>
    <committed>10737418240</committed>
    <storage_format>v1</storage_format>
    <host id="0656f432-923a-11e0-ad20-5254004ac988"
      href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988">
</storage_domain>
```

3.8. EXAMPLE: CREATE NFS ISO STORAGE

An NFS ISO storage domain is a mounted NFS share attached to a data center and provides storage for

DVD/CD-ROM ISO and virtual floppy disk (VFD) image files. Creation of a new storage domain requires a **POST** request, with the storage domain representation included, sent to the URL of the storage domain collection.

You can enable the wipe after delete option by default on the storage domain. To configure this specify **<wipe_after_delete>** in the **POST** request. This option can be edited after the domain is created, but doing so will not change the wipe after delete property of disks that already exist.

Example 3.8. Create an NFS ISO storage domain

Request:

```
POST /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
  <name>iso1</name>
  <type>iso</type>
  <storage>
    <type>nfs</type>
    <address>192.168.0.10</address>
    <path>/iso1</path>
  </storage>
  <host>
    <name>hypervisor</name>
  </host>
</storage_domain>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<storage_domain><name>iso1</name><type>iso</type> \
    <storage><type>nfs</type><address>192.168.0.10</address> \
    <path>/iso1</path></storage> \
    <host><name>hypervisor</name></host></storage_domain>" \
    https://[RHEVM Host]:443/ovirt-engine/api/storagedomains
```

The API creates a NFS iso storage domain called **iso1** with an export path of **192.168.0.10:/iso1** and gets access to the storage domain through the **hypervisor** host. The API also returns the following representation of the newly created storage domain resource.

Result:

```
HTTP/1.1 200 OK
Accept: application/xml

<storage_domain id="00f0d9ce-da15-4b9e-9e3e-3c898fa8b6da"
  href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da">
    <name>iso1</name>
    <link rel="permissions"
      href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
```

```
3c898fa8b6da/
      permissions"/>
    <link rel="files"
      href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/files"/>
    <type>iso</type>
    <host id="" href="">
    <master>false</master>
    <storage>
        <type>nfs</type>
        <address>192.168.0.10</address>
        <path>/iso1</path>
    </storage>
    <available>82678120448</available>
    <used>18253611008</used>
    <committed>0</committed>
    <storage_format>v1</storage_format>
    <host id="0656f432-923a-11e0-ad20-5254004ac988"
      href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988">
</storage_domain>
```

3.9. EXAMPLE: ATTACH STORAGE DOMAINS TO DATA CENTER

The following example attaches the **data1** and **iso1** storage domains to the **Default** data center.

Example 3.9. Attach data1 storage domain to the Default data center

Request:

```
POST /ovirt-engine/api/datacenters/01a45ff0-915a-11e0-8b87-
5254004ac988/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
  <name>data1</name>
</storage_domain>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<storage_domain><name>data1</name></storage_domain>" \
    https://[RHEVM Host]:443/ovirt-engine/api/datacenters/01a45ff0-915a-
11e0-8b87-5254004ac988/storagedomains
```

Example 3.10. Attach iso1 storage domain to the Default data center

Request:

```
POST /ovirt-engine/api/datacenters/01a45ff0-915a-11e0-8b87-
5254004ac988/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
  <name>iso1</name>
</storage_domain>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<storage_domain><name>iso1</name></storage_domain>" \
    https://[RHEVM Host]:443/ovirt-engine/api/datacenters/01a45ff0-915a-
11e0-8b87-5254004ac988/storagedomains
```

These **POST** requests place our two new **storage_domain** resources in the **storagedomains** sub-collection of the **Default** data center. This means the **storagedomains** sub-collection contains attached storage domains of the data center.

3.10. EXAMPLE: ACTIVATE STORAGE DOMAINS

This example activates the **data1** and **iso1** storage domains for the Red Hat Virtualization Manager's use.

Example 3.11. Activate data1 storage domain

Request:

```
POST /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/
9ca7cb40-9a2a-4513-acef-dc254af57aac/activate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<action/>" \
    https://[RHEVM Host]:443/ovirt-engine/api/datacenters/d70d5e2d-b8ad-
494a-a4d2-c7a5631073c4/storagedomains/9ca7cb40-9a2a-4513-acef-
dc254af57aac/activate
```

Example 3.12. Activate iso1 storage domain

Request:

```
POST /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/
00f0d9ce-da15-4b9e-9e3e-3c898fa8b6da/activate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<action/>"
    https://[RHEVM Host]:443/ovirt-engine/api/datacenters/d70d5e2d-b8ad-
494a-a4d2-c7a5631073c4/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/activate
```

This activates both storage domains for use with the data center.

3.11. EXAMPLE: CREATE VIRTUAL MACHINE

The following example creates a virtual machine called **vm1** on the **Default** cluster using the virtualization environment's **Blank** template as a basis. The request also defines the virtual machine's **memory** as 512 MB and sets the **boot** device to a virtual hard disk.

Example 3.13. Create a virtual machine

Request:

```
POST /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <name>vm1</name>
    <cluster>
        <name>default</name>
    </cluster>
    <template>
        <name>Blank</name>
    </template>
    <memory>536870912</memory>
    <os>
        <boot dev="hd"/>
    </os>
    <cpu_profile id="0000001a-001a-001a-001a-00000000035e"/>
</vm>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] --cacert [CERT] -d "<vm>
<name>vm1</name><cluster><name>default</name></cluster><template>
<name>Blank</name></template><memory>536870912</memory><os><boot
dev='hd'/></os><cpu_profile id='0000001a-001a-001a-001a-00000000035e'/>
</vm>" https://[RHEVM Host]:443/ovirt-engine/api/vms
```

Result:

```
HTTP/1.1 200 OK
Accept: application/xml

<vm id="6efc0cfa-8495-4a96-93e5-ee490328cf48"
  href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-ee490328cf48">
    <name>vm1</name>
    <actions>
        <link rel="shutdown"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/shutdown"/>
        <link rel="start"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/start"/>
        <link rel="stop"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/stop"/>
        <link rel="reboot"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/reboot"/>
        <link rel="suspend"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/suspend"/>
        <link rel="detach"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/detach"/>
        <link rel="export"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/export"/>
        <link rel="move"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/move"/>
        <link rel="ticket"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/ticket"/>
        <link rel="migrate"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/migrate"/>
        <link rel="undo_snapshot"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/undo_snapshot"/>
        <link rel="commit_snapshot"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/commit_snapshot"/>
        <link rel="preview_snapshot"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/preview_snapshot"/>
```

```
        <link rel="logon"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/logon"/>
        <link rel="cancelmigration"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/cancelmigration"/>
        <link rel="maintenance"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/maintenance"/>
        <link rel="clone"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/clone"/>
    </actions>
    <link rel="applications"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/applications"/>
    <link rel="disks"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/disks"/>
    <link rel="nics"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/nics"/>
    <link rel="cdroms"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/cdroms"/>
    <link rel="snapshots"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/snapshots"/>
    <link rel="tags"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/tags"/>
    <link rel="permissions"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/permissions"/>
    <link rel="statistics"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/statistics"/>
    <link rel="reporteddevices"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/reporteddevices"/>
    <link rel="watchdogs"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/watchdogs"/>
    <link rel="sessions"
      href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48/sessions"/>
    <type>desktop</type>
    <status>
        <state>down</state>
    </status>
    <memory>536870912</memory>
    <cpu>
        <topology cores="1" sockets="1"/>
    </cpu>
    <os type="Unassigned">
        <boot dev="cdrom"/>
```

```
    </os>
    <high_availability>
        <enabled>false</enabled>
        <priority>0</priority>
    </high_availability>
    <display>
        <type>spice</type>
        <monitors>1</monitors>
        <single_qxl_pci>false</single_qxl_pci>
        <allow_override>false</allow_override>
        <smartcard_enabled>false</smartcard_enabled>
        <file_transfer_enabled>true</file_transfer_enabled>
        <copy_paste_enabled>true</copy_paste_enabled>
    </display>
    <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"
      href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95"/>
    <template id="00000000-0000-0000-0000-000000000000"
      href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000"/>
    <stop_time>2011-06-15T04:48:02.167Z</stop_time>
    <creation_time>2011-06-15T14:48:02.078+10:00</creation_time>
    <origin>rhev</origin>
    <stateless>false</stateless>
    <delete_protected>false</delete_protected>
    <sso>
        <methods>
            <method id="GUEST_AGENT"/>
        </methods>
    </sso>
    <console enabled="false"/>
    <timezone>Etc/GMT</timezone>
    <initialization>
        <configuration>
            <type>ovf</type>
            <data>...</data>
        </configuration>
    </initialization>
    <placement_policy>
        <affinity>migratable</affinity>
    </placement_policy>
    <memory_policy>
        <guaranteed>536870912</guaranteed>
        <ballooning>true</ballooning>
    </memory_policy>
    <usb>
        <enabled>false</enabled>
    </usb>
    <soundcard_enabled>true</soundcard_enabled>
    <migration_downtime>-1</migration_downtime>
    <virtio_scsi enabled="true"/>
    <cpu_profile id="0000001a-001a-001a-001a-00000000035e"/>
    <next_run_configuration_exists>false</next_run_configuration_exists>
    <numa_tune_mode>interleave</numa_tune_mode>
</vm>
```

3.12. EXAMPLE: CREATE VIRTUAL MACHINE NIC

The following example creates a virtual network interface to connect the example virtual machine to the **ovirtmgmt** network.

Example 3.14. Create a virtual machine NIC

Request:

```
POST /ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-ee490328cf48/nics
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<nic>
  <interface>virtio</interface>
  <name>nic1</name>
  <network>
    <name>ovirtmgmt</name>
  </network>
</nic>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<nic><name>nic1</name><network><name>ovirtmgmt</name></network>
</nic>" \
    https://[RHEVM Host]:443/ovirt-engine/api/vms/6efc0cfa-8495-4a96-
93e5-ee490328cf48/nics
```

3.13. EXAMPLE: CREATE VIRTUAL MACHINE STORAGE DISK

The following example creates an 8 GB Copy-On-Write storage disk for the example virtual machine.

Example 3.15. Create a virtual machine storage disk

Request:

```
POST /ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-ee490328cf48/disks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <storage_domains>
        <storage_domain id="9ca7cb40-9a2a-4513-acef-dc254af57aac"/>
    </storage_domains>
    <size>8589934592</size>
    <type>system</type>
    <interface>virtio</interface>
```

```
    <format>cow</format>
    <bootable>true</bootable>
</disk>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<disk><storage_domains> \
    <storage_domain id='9ca7cb40-9a2a-4513-acef-dc254af57aac'/> \
    </storage_domains><size>8589934592</size><type>system</type> \
    <interface>virtio</interface><format>cow</format> \
    <bootable>true</bootable></disk>" \
    https://[RHEVM Host]:443/ovirt-engine/api/vms/6efc0cfa-8495-4a96-
93e5-ee490328cf48/disks
```

The **storage_domain** element tells the API to store the disk on the **data1** storage domain.

3.14. EXAMPLE: ATTACH ISO IMAGE TO VIRTUAL MACHINE

The boot media for our example virtual machine requires an CD-ROM or DVD ISO image for an operating system installation. This example uses a Red Hat Enterprise Server 6 ISO image for installation.

ISO images must be available in the **iso1** ISO domain for the virtual machines to use. Red Hat Virtualization Platform provides an uploader tool that ensures that the ISO images are uploaded into the correct directory path with the correct user permissions.

Once the ISO is uploaded, an API user requests the ISO storage domain's **files** sub-collection to view the file resource:

Example 3.16. View the files sub-collection in an ISO storage domain

Request:

```
GET /ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/files HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert
[CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/storagedomains/00f0d9ce-
da15-4b9e-9e3e-3c898fa8b6da/files
```

The API returns the following representation of the files sub-collection:

```
<files>
    <file id="rhel-server-6.0-x86_64-dvd.iso"
      href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
```

```
3c898fa8b6da/
      files/rhel-server-6.0-x86_64-dvd.iso.iso">
        <name>rhel-server-6.0-x86_64-dvd.iso.iso</name>
        <storage_domain id="00f0d9ce-da15-4b9e-9e3e-3c898fa8b6da"
          href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-
9e3e-3c898fa8b6da"/>
    </file>
</files>
```

An API user attaches the **rhel-server-6.0-x86_64-dvd.iso** to our example virtual machine. Attaching an ISO image is equivalent to using the **Change CD** button in the Administration or User Portal.

Example 3.17. Attach an ISO image to the virtual machine

Request:

```
POST /ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-ee490328cf48/cdroms
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cdrom>
  <file id="rhel-server-6.0-x86_64-dvd.iso"/>
</cdrom>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<cdrom><file id='rhel-server-6.0-x86_64-dvd.iso'/></cdrom>" \
    https://[RHEVM Host]:443/ovirt-engine/api/vms/6efc0cfa-8495-4a96-
93e5-ee490328cf48/cdroms
```

3.15. EXAMPLE: START VIRTUAL MACHINE

The virtual environment is complete and the virtual machine contains all necessary components to function. This example starts the virtual machine using the **start** action.

Example 3.18. Start the virtual machine

Request:

```
POST /ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-ee490328cf48/start
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
  <vm>
```

```
    <os>
      <boot dev="cdrom"/>
    </os>
  </vm>
</action>
```

cURL command:

```
# curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" \
    -u [USER:PASS] --cacert [CERT] \
    -d "<action><vm><os><boot dev='cdrom'/></os></vm></action>" \
    https://[RHEVM Host]:443/ovirt-engine/api/vms/6efc0cfa-8495-4a96-
93e5-ee490328cf48/start
```

The additional message entity sets the virtual machine's boot device to CD-ROM for this boot only. This enables the virtual machine to install Red Hat Enterprise Server 6 from the attached ISO image. The boot device reverts back to **disk** for all future boots.

3.16. EXAMPLE: CHECK SYSTEM EVENTS

The **start** action for the **vm1** creates several entries in the **events** collection. This example lists the events collection and identifies events specific to the API starting a virtual machine.

Example 3.19. List the events collection

Request:

```
GET /ovirt-engine/api/events HTTP/1.1
Accept: application/xml
```

cURL command:

```
# curl -X GET -H "Accept: application/xml" -u [USER:PASS] \
    --cacert [CERT] \
    https://[RHEVM Host]:443/ovirt-engine/api/events
```

Result:

```
<events>
    ...
    <event id="103" href="/ovirt-engine/api/events/103">
        <description>User admin logged out.</description>
        <code>31</code>
        <severity>normal</severity>
        <time>2011-06-29T17:42:41.544+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
    </event>
    <event id="102" href="/ovirt-engine/api/events/102">
        <description>vm1 was started by admin (Host: hypervisor).
```

```
</description>
        <code>153</code>
        <severity>normal</severity>
        <time>2011-06-29T17:42:41.499+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
        <vm id="6efc0cfa-8495-4a96-93e5-ee490328cf48"
          href="/ovirt-engine/api/vms/6efc0cfa-8495-4a96-93e5-
ee490328cf48"/>
        <host id="0656f432-923a-11e0-ad20-5254004ac988"
          href="/ovirt-engine/api/hosts/0656f432-923a-11e0-ad20-
5254004ac988"/>
    </event>
    <event id="101" href="/ovirt-engine/api/events/101">
        <description>User admin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2011-06-29T17:42:40.505+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
    </event>
    ...
</events>
```

The following events occur:

- **id="101"** - The API authenticates with the **admin** user's user name and password.
- **id="102"** - The API, acting as the **admin** user, starts **vm1** on the **hypervisor** host.
- **id="103"** - The API logs out of the **admin** user account.

CHAPTER 4. ENTRY POINT

A user begins interacting with the API through a **GET** request on the entry point URI consisting of a **host** and **base**.

Example 4.1. Accessing the API Entry Point

If the **host** is **www.example.com** and the **base** is **/ovirt-engine/api**, the entry point appears with the following request:

```
GET /ovirt-engine/api HTTP/1.1
Accept: application/xml
Host: www.example.com
Authorization: [base64 encoded credentials]

HTTP/1.1 200 OK
Content-Type: application/xml

<api>
    <link rel="hosts" href="/ovirt-engine/api/hosts"/>
    <link rel="vms" href="/ovirt-engine/api/vms"/>
    ...
    <product_info>
        <name>Red Hat Virtualization</name>
        <vendor>Red Hat</vendor>
        <version revision="0" build="0" minor="0" major="4"/>
    </product_info>
    <special_objects>
        <link rel="templates/blank" href="..."/>
        <link rel="tags/root" href="..."/>
    </special_objects>
    <summary>
        <vms>
            <total>10</total>
            <active>3</active>
        </vms>
        <hosts>
            <total>2</total>
            <active>2</active>
        </hosts>
        <users>
            <total>8</total>
            <active>2</active>
        </users>
        <storage_domains>
            <total>2</total>
            <active>2</active>
        </storage_domains>
    </summary>
</ovirt-engine/api>
```

NOTE

For simplicity, all other examples omit the **`Host:`** and **`Authorization:`** request headers and assume the **`base`** is the default **`/ovirt-engine/api`** path. This base path differs depending on your implementation.

4.1. PRODUCT INFORMATION

The entry point contains a **`product_info`** element to help an API user determine the legitimacy of the Red Hat Virtualization environment. This includes the **`name`** of the product, the **`vendor`** and the **`version`**.

Example 4.2. Verify a genuine Red Hat Virtualization environment

The follow elements identify a genuine Red Hat Virtualization 4.0 environment:

```
<api>
    ...
    <product_info>
        <name>Red Hat Virtualization</name>
        <vendor>Red Hat</vendor>
    <version>
      <build>2</build>
      <full_version>4.0.2.3-0.1.el7ev</full_version>
      <major>4</major>
      <minor>0</minor>
      <revision>0</revision>
    </version>
    </product_info>
    ...
</ovirt-engine/api>
```

4.2. LINK ELEMENTS

Access to the Entry Point provides **`link`** elements and URIs for all of the resource collections the API exposes. Each collection uses a relation type to identify the URI a client needs.

Table 4.1. Available Relationship Types

Relationship	Description
`capabilities`	Supported capabilities of the Red Hat Virtualization Manager.
`datacenters`	Data centers.
`clusters`	Host clusters.
`networks`	Virtual networks.
`storagedomains`	Storage domains.

Relationship	Description
`hosts`	Hosts.
`vms`	Virtual machines.
`disks`	Virtual disks.
`templates`	Templates.
`vmpools`	Virtual machine pools.
`domains`	Identity service domains.
`groups`	Imported identity service groups.
`roles`	Roles.
`users`	Users.
`tags`	Tags.
`events`	Events.

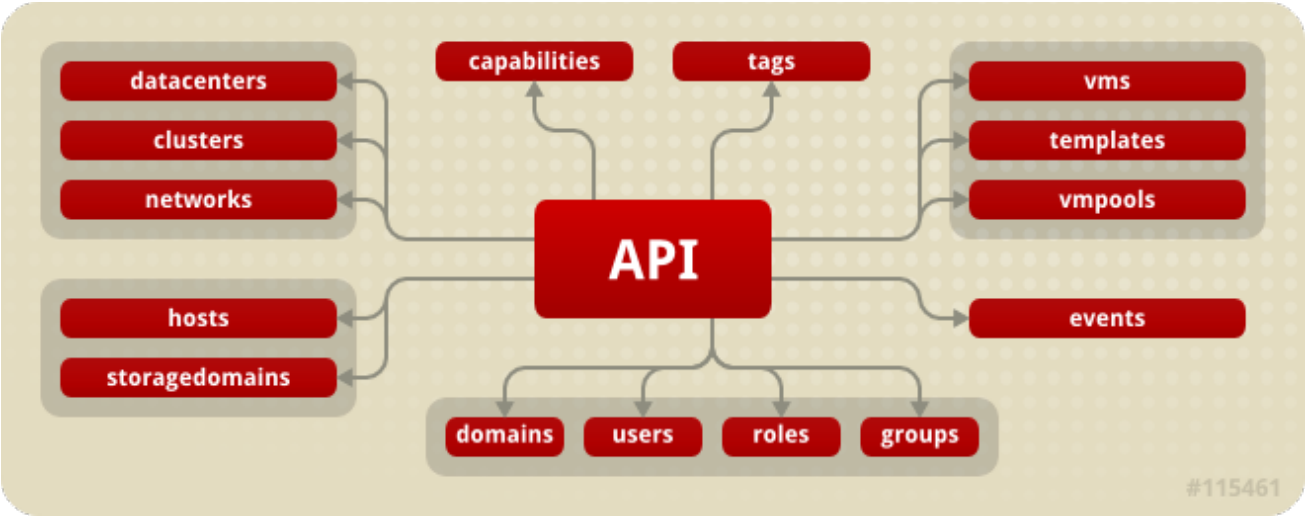
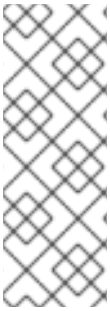


Figure 4.1. The relationship between the API entry point and the resource collections exposed by the API



NOTE

All URIs shown in example responses are illustrative. The format of all URIs returned by the server is opaque. Clients navigate to specific resources through the entry point URI and use the relationship types to access the URIs.

The server chooses to include absolute URIs or absolute paths [3] in the **`link`** element's **`href`** attribute, so clients are required to handle either form.

The **`link`** elements also contain a set of **`search`** URIs for certain collections. These URIs use URI templates [4] to integrate search queries. The purpose of the URI template is to accept a search expression using the natural HTTP pattern of a query parameter. The client does not require prior knowledge of the URI structure. Thus clients should treat these templates as being opaque and access them with a URI template library.

Each search query URI template is identified with a relation type using the convention **`"collection/search"`**.

Table 4.2. Relationships associated with search query URIs

Relationship	Description
`datacenters/search`	Query data centers.
`clusters/search`	Query host clusters.
`storagedomains/search`	Query storage domains.
`hosts/search`	Query hosts.
`vms/search`	Query virtual machines.
`disks/search`	Query disks.
`templates/search`	Query templates.
`vmpools/search`	Query virtual machine pools.
`events/search`	Query events.
`users/search`	Query users.

4.3. SPECIAL OBJECT ELEMENTS

Special object elements define relationships to special fixed resources within the virtualization environment.

Table 4.3. Special Objects

Relationship	Description
templates/blank	The default **blank** virtual machine template for your virtualization environment. This template exists in every cluster as opposed to a standard template, which only exists in a single cluster.
tags/root	The **root** tag that acts as a base for tag hierarchy in your virtualization environment.

4.4. SUMMARY ELEMENT

The summary element shows a high level summary of the system's statistics.

Table 4.4. Summary Elements

Element	Description
vms	Total number of vms and total number of active vms.
hosts	Total number of hosts and total number of active hosts.
users	Total number of users and total number of active users.
storage_domains	Total number of storage domains and total number of active storage domains.

4.5. RESTFUL SERVICE DESCRIPTION LANGUAGE (RSDL)

RESTful Service Description Language (RSDL) provides a description of the structure and elements in the REST API in one whole XML specification. Invoke the RSDL using the following request.

```
GET /ovirt-engine/api?rsdl HTTP/1.1
Accept: application/xml
```

This produces an XML document in the following format:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rsdl href="/ovirt-engine/api?rsdl" rel="rsdl">
    <description>...</description>
    <version major="4" minor="0" build="0" revision="0"/>
    <schema href="/ovirt-engine/api?schema" rel="schema">
        <name>...</name>
        <description>...</description>
    </schema>
    <links>
        <link href="/ovirt-engine/api/capabilities" rel="get">
            ...
        </link>
```

```
        ...
    </links>
</rsdl>
```

Table 4.5. RSDL Structure Elements

Element	Description
description	A plain text description of the RSDL document.
version	The API version, including **major** release, **minor** release, **build** and **revision**.
schema	A link to the XML schema (XSD) file.
links	Defines each **link** in the API.

Each **link** element contains the following a structure:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rsdl href="/ovirt-engine/api?rsdl" rel="rsdl">
    ...
    <links>
        <link href="/ovirt-engine/api/..." rel="...">
            <request>
                <http_method>...</http_method>
                <headers>
                    <header>
                        <name>...</name>
                        <value>...</value>
                    </header>
                    ...
                </headers>
                <body>
                    <type>...</type>
                    <parameters_set>
                        <parameter required="..." type="...">
                            <name>...</name>
                        </parameter>
                        ...
                    </parameters_set>
                </body>
            </request>
            <response>
                <type>...</type>
            </response>
        </link>
        ...
    </links>
</rsdl>
```

Table 4.6. RSDL Link Structure Elements

Element	Description
link	A URI for API requests. Includes a URI attribute (**href**) and a relationship type attribute (**rel**).
request	Defines the request properties required for the link.
http_method	The method type to access this link. Includes the standard HTTP methods for REST API access: **GET**, **POST**, **PUT** and **DELETE**.
headers	Defines the headers for the HTTP request. Contains a series of **header** elements, which each contain a header **name** and **value** to define the header.
body	Defines the body for the HTTP request. Contains a resource **type** and a **parameter_set**, which contains a sets of **parameter** elements with attributes to define whether they are **required** for a request and the data **type**. The **parameter** element also includes a **name** element to define the Red Hat Virtualization Manager property to modify and also a further **parameter_set** subset if **type** is set to **collection**.
response	Defines the output for the HTTP request. Contains a **type** element to define the resource structure to output.

Use the RSDL in your applications as a method to map all links and parameter requirements for controlling a Red Hat Virtualization environment.

4.6. RED HAT VIRTUALIZATION WINDOWS GUEST VSS SUPPORT

The Red Hat Virtualization Backup and Restore API provides integration with Microsoft Windows Volume Shadow Copy Service (VSS) using **qemu-ga**. The VSS provider registration is made in the guest level as part of the Guest Tools deployment.

qemu-ga provides VSS support and live snapshots attempt to quiesce whenever possible.

4.7. QEMU GUEST AGENT OVERVIEW

In Red Hat Enterprise Linux 6.4, the QEMU Guest Agent (QEMU GA) provided protection against the corruption of Linux guest virtual machines. Before issuing a snapshot request or creating a backup copy of the disk, the management stack (libvirt) sent a **guest-fsfreeze-freeze QMP** command to the QEMU GA via the virtio-serial port. This command caused the guest agent to freeze all of the guest virtual machine's filesystems, via the **FIFREEZE ioctl()** kernel function. This **ioctl()** function is implemented by the Linux kernel in the guest virtual machine. The function flushes the filesystem cache in the guest virtual machine's kernel, brings the filesystem into a consistent state, and denies all userspace threads write access to the filesystem.

Only after the **QEMU GA** reported success, **libvirt** would proceed with the snapshot. At its completion, **libvirt** sends the **guest-fsfreeze-thaw QMP** command to the **QEMU GA** over the virtio-serial port. This command tells the **QEMU GA** to issue a **FITHAW ioctl()**, which unblocks the userspace threads that were previously denied write access, and resumes normal processing. This process did not ensure that application-level data was in a consistent state when the virtual disk snapshot was taken. This was

evident in cases where the fsck utility found no problems on filesystems restored from snapshots, and yet applications were not able to resume processing from the point where the snapshot was taken and userspace processes may not have written their internal buffers to files on the disk.

Red Hat Enterprise Linux 6.5 ensures that both file and application-level synchronization (flushing) are done. Guest system administrators can write and install application-specific freezing and thawing hook scripts. Before freezing the filesystems, the **QEMU GA** invokes the main hook script (included in the **QEMU GA** package). The main hook script in turn calls individual application-specific scripts, prepared by the guest system administrators, that temporarily deactivate all guest virtual machine applications. All of these actions occur when the mode is changed to "freeze".

Just before filesystems are frozen, the guest system administrator's scripts cause the databases and other file system applications to flush their working buffers to the virtual disk and to stop accepting further client connections. The applications then bring their data files into a consistent state where resumption of processing, with the reactivated (or a freshly started) instance of the application (after restoring the virtual disk from backup) is possible. When all scripts are done making their respective applications inactive, and the main hook script returns, **QEMU GA** proceeds to freeze filesystems, and the management stack takes the snapshot. Once all this is done, and it is confirmed that the snapshot is taken, the file system will resume to serve write requests. This process is called thawing.

Thawing is freezing in reverse order. Instructed by **libvirt**, **QEMU GA** thaws the guest virtual machine's filesystems. It then invokes individual hook scripts (via the main hook script) to resume or restart applications that had been inactivated during the freeze process.

4.8. VSS TRANSACTION FLOW

In processing a backup, the requester and the writers coordinate to do several things: to provide a stable system image from which to back up data (the shadow copied volume), to group files together on the basis of their usage, and to store information on the saved data. This must all be done with minimal interruption of the writer's normal work flow.

A requester (in our case the Backup Vendor) queries writers for their metadata, processes this data, notifies the writers prior to the beginning of the shadow copy and of the backup operations, and then notifies the writers again after the shadow copy and backup operations end.

Here is how the QEMU VSS provider is registered in Windows OS after the Guest Tools installation:

```
C:\Users\Administrator>vssadmin list providers
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line tool
(C) Copyright 2001-2005 Microsoft Corp.

Provider name: 'QEMU Guest Agent VSS Provider'
   Provider type: Software
   Provider Id: {3629d4ed-ee09-4e0e-9a5c-6d8ba2872aef}
   Version: 0.12.1
```

[3] The RFC describing Uniform Resource Locator Generic Syntax provides a Collected ABNF for URI that explains the difference between these forms.

[4] The Internet-Draft describing the format of a URI Template is available at http://tools.ietf.org/html/draft-gregorio-uritemplate-03.

CHAPTER 5. COMPATIBILITY LEVEL VERSIONS

Each host connected to Red Hat Virtualization Manager contains a version of VDSM. VDSM is the agent within the virtualization infrastructure that runs on a hypervisor or host and provides local management for virtual machines, networks and storage. Red Hat Virtualization Manager controls hypervisors and hosts using current or earlier versions of VDSM.

The Manager migrates virtual machines from host to host within a cluster. This means the Manager excludes certain features from a current version of VDSM until all hosts within a cluster have the same VDSM version, or more recent, installed.

The API represents this concept as a **`compatibility level`** for each host, corresponding to the version of VDSM installed. A **`version`** element contains **`major`** and **`minor`** attributes, which describe the compatibility level.

When an administrator upgrades all hosts within a cluster to a certain level, the **`version`** level appears under a **`supported_versions`** element. This indicates the cluster's **`version`** is now updatable to that level. Once the administrator updates all clusters within a data center to a given level, the data center is updatable to that level.

5.1. UPGRADING COMPATIBILITY LEVELS

Example 5.1. Upgrading compatibility levels

The API reports the following compatibility levels for Red Hat Enterprise Virtualization Manager 3.4 instance:

```
<host ...>
    ...
    <version major="4" minor="14" build="11" revision="0"
full_version="vdsm-4.14.11-5.el6ev"/>
    ...
</host>

<cluster ...>
    ...
    <version major="3" minor="4"/>
    ...
</cluster>

<data_center ...>
    ...
    <version major="3" minor="4"/>
    </supported_versions>
    ...
</data_center>
```

All hosts within a cluster are updated to VDSM **`3.5`** and the API reports:

```
<host ...>
    ...
    <version major="4" minor="16" build="7" revision="4"
full_version="vdsm-4.16.7.4-1.el6ev"/>
```

```
    ...
</host>

<cluster ...>
    ...
    <version major="3" minor="4"/>
    <supported_versions>
        <version major="3" minor="5"/>
    </supported_versions>
    ...
</cluster>

<data_center ...>
    ...
    <version major="3" minor="4"/>
    <supported_versions/>
    ...
</data_center>
```

The cluster is now updatable to **3.5**. When the cluster is updated, the API reports:

```
<cluster ...>
    ...
    <version major="3" minor="5"/>
    <supported_versions/>
    ...
</cluster>

<data_center ...>
    ...
    <version major="3" minor="4"/>
    <supported_versions>
        <version major="3" minor="5"/>
    </supported_versions>
    ...
</data_center>
```

The API user updates the data center to **3.5**. Once upgraded, the API exposes features available in Red Hat Enterprise Virtualization 3.5 for this data center.

CHAPTER 6. CAPABILITIES

The **capabilities** collection provides information about the capabilities that versions of Red Hat Virtualization support. These capabilities include active features and available enumerated values for specific properties.

To retrieve a full list of the capabilities for all versions of Red Hat Enterprise Virtualization from 3.2 to the latest version, submit the following request:

```
GET /ovirt-engine/api/capabilities/ HTTP/1.1
Content-Type: application/xml
Accept: application/xml
```

6.1. VERSION-DEPENDENT CAPABILITIES

The **capabilities** element contains any number of **version** elements that describe capabilities dependent on a compatibility level.

The **version** element includes attributes for **major** and **minor** version numbers. This indicates the current version level.

The following representation shows capabilities specific to Red Hat Enterprise Virtualization Manager **3.5**, **3.6**, and **4.0** respectively:

```
<capabilities>
    <version major="3" minor="5">
        ...
    </version>
    <version major="3" minor="6">
        ...
    </version>
    <version major="4" minor="0">
        ...
    </version>
    ...
</capabilities>
```

Each **version** contains a series of capabilities dependent on the version specified.

6.2. CURRENT VERSION

The **current** element signifies if the **version** specified is the most recent supported compatibility level. The value is a Boolean **true** or **false**.

```
<capabilities>
    <version major="4" minor="0">
        ...
        <current>true</current>
        ...
    </version>
</capabilities>
```

6.3. FEATURES

Each version contains a list of compatible features. The following table lists the features compatible with Red Hat Virtualization 4.0.

Table 6.1. Feature Types

Feature	Description
Transparent huge pages memory policy	Allows you to define the availability of transparent huge pages for hosts. Acceptable values are **`true`** or **`false`**.
Gluster support	This features provides support for using Gluster Volumes and Bricks as storage.
POSIX-FS storage type	This feature provides support for the POSIX-FS storage type.
Port mirroring	Allows you to define the availability of port mirroring for virtual network interface cards. Acceptable values are **`true`** or **`false`**.
Display server time	Displays the current date and time in the API.
Display host memory	Displays the total memory for a specific host.
Display host sockets	Allows you to define the topology of a host CPU. Takes three attributes - **`sockets`**, **`threads`** and **`cores`** - which define the number of host sockets displayed, the number of threads and the number of cores per socket.
Search case sensitivity	Allows you to specify whether a search query is case sensitive by providing the **`case-sensitive=`*`true`*`|`*`false`*** URL parameter.
Maximum results for **`GET`** requests	Allows you to specify the maximum number of results returned from a **`GET`** request.
JSON content type	Allows you to define a header that makes it possible to set a correlation ID for **`POST`** and **`PUT`** requests.
Activate and deactivate disks	Allows you to activate or deactivate a disk by specifying **`activate`** or **`deactivate`** as an action on a specific virtual disk.
Activate and deactivate network interface cards	Allows you to activate or deactivate a network interface card by specifying **`activate`** or **`deactivate`** as an action on a specific network interface card.
Snapshot refactoring	Allows you to refactor snapshots for virtual machines.

Feature	Description
Remove template disks from specified storage domain	Allows you to remove virtual machine template disks from a specific storage domain using a **DELETE** request.
Floating disks	Floating disks are disks that are not attached to any virtual machine. With this feature, such disks also appear in the root collection rather than under specific virtual machines.
Asynchronous deletion	Allows you to specify that **DELETE** requests are to be performed asynchronously by specifying the **async** URL parameter.
Session-based authentication	Allows you to maintain a client-server session by providing an appropriate header, eliminating the need to log in with each request.
Virtual machine applications	Allows you to view a list of applications installed on a specific virtual machine. This list is located in the **applications** element of a specific virtual machine.
VirtIO-SCSI support	This feature provides support for para-virtualized SCSI controller devices.
Custom resource comments	Allows you to add custom comments to data centers and other resources.
Refresh host capabilities	Allows you to synchronize data on hosts and refresh the list of network interfaces available to a specific host.
Memory snapshot	Allows you to include the memory state as part of a virtual machine snapshot.
Watchdog device	Allows you to create watchdog devices for virtual machines.
SSH authentication method	Allows you to authenticate with hosts over SSH using an administrative user password or SSH public key.
Force select SPM	Allows you to force the selection of a host as SPM.
Console device	Allows you to control the attachment of console devices in virtual machines.
Storage server connections for storage domains	Allows you to view storage server connections to or from a specific storage domain.
Attach and detach storage server connections	Allows you to attach or detach storage server connections to or from a specific storage domain.

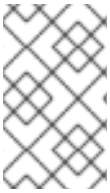
Feature	Description
Single PCI for Qxl	Allows you to view multiple video devices via a single PCI guest device.
Add virtual machine from OVF configuration	Allows you to add a virtual machine from a provided OVF configuration.
Virtual network interface card profiles	Allows you to configure a profile that defines quality of service, custom properties and port mirroring for a specific virtual network interface card.
Image storage domains (tech preview)	Allows you to import images from and export images to an image storage domain such as an OpenStack image service (Glance).
Virtual machine fully qualified domain names	Allows you to retrieve the fully qualified domain name of a specific virtual machine.
Attaching disk snapshots to virtual machines	This feature provides support for attaching disk snapshots to virtual machines.
Cloud-Init	Allows you to initialize a virtual machine using Cloud Init.
Gluster brick management	Allows you to delete gluster bricks with data migration using the actions **`migrate`** and **`DELETE`**. The **`migrate`** action and **`stopmigrate`** action allow you to migrate data and reuse the brick.
Copy and move back-end disks	Allows you to copy and move disks in additional contexts.
Network labels	Allows you to provision networks on hosts using labels.
Reboot virtual machines	Allows you to reboot virtual machines via a single action.

A full list of feature elements and their attributes is located at the top of the section for the relevant version:

```
<capabilities>
    <version major="4" minor="0">
        ...
        <features>
            <feature>
             <name>Transparent-Huge-Pages Memory Policy</name>
             <transparent_huepages/>
            </feature>
        </features>
        ...
    </version>
</capabilities>
```

CHAPTER 7. COMMON FEATURES

7.1. ELEMENT PROPERTY ICONS



NOTE

Throughout this guide, the elements of each resource are detailed in tables. These tables include a properties column, displaying icons depicting element properties. The meaning of these icons is shown in Table 7.1, “Element property icons”.

Table 7.1. Element property icons

Property	Description	Icon
Required for creation	These elements must be included in the client-provided representation of a resource on creation, but are not mandatory for an update of a resource.	
Non-updatable	These elements cannot have their value changed when updating a resource. Include these elements in a client-provided representation on update only if their values are not altered by the API user. If altered, the API reports an error.	
Read-only	These elements are read-only. Values for read-only elements are not created or modified.	

7.2. REPRESENTATIONS

7.2.1. Representations

The API structures resource representations in the following XML document structure:

```
<resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id">
    <name>Resource-Name</name>
    <description>A description of the resource</description>
    ...
</resource>
```

In the context of a virtual machine, the representation appears as follows:

```
<vm id="5b9bbce5-0d72-4f56-b931-5d449181ee06"
  href="/ovirt-engine/api/vms/5b9bbce5-0d72-4f56-b931-5d449181ee06">
    <name>RHEL6-Machine</name>
    <description>Red Hat Enterprise Linux 6 Virtual Machine</description>
    ...
</vm>
```

7.2.2. Common Attributes to Resource Representations

All resource representations contain a set of common attributes

Table 7.2. Common attributes to resource representations

Attribute	Type	Description	Properties
id	GUID	Each resource in the virtualization infrastructure contains an **id**, which acts as a globally unique identifier (GUID). The GUID is the primary method of resource identification.	
href	string	The canonical location of the resource as an absolute path.	

7.2.3. Common Elements to Resource Representations

All resource representations contain a set of common elements.

Table 7.3. Common elements to resource representations

Element	Type	Description	Properties
name	string	A user-supplied human readable name for the resource. The **name** is unique across all resources of its type.	
description	string	A free-form user-supplied human readable description of the resource.	

7.3. COLLECTIONS

7.3.1. Collections

A collection is a set of resources of the same type. The API provides both top-level collections and sub-collections. An example of a top-level collection is the **hosts** collection which contains all virtualization hosts in the environment. An example of a sub-collection is the **host.nics** collection which contains resources for all network interface cards attached to a host resource.

7.3.2. Listing All Resources in a Collection

Obtain a listing of resources in a collection with a **GET** request on the collection URI obtained from the entry point.

Include an **Accept** HTTP header to define the MIME type for the response format.

```
GET /ovirt-engine/api/[collection] HTTP/1.1
Accept: [MIME type]
```

7.3.3. Listing Extended Resource Sub-Collections

The API extends collection representations to include sub-collections when the **Accept** header includes the **detail** parameter.

```
GET /ovirt-engine/api/collection HTTP/1.1
Accept: application/xml; detail=subcollection
```

This includes multiple sub-collection requests using either separated **detail** parameters:

```
GET /ovirt-engine/api/collection HTTP/1.1
Accept: application/xml; detail=subcollection1; detail=subcollection2
```

Or one **detail** parameter that separates the sub-collection with the **+** operator:

```
GET /ovirt-engine/api/collection HTTP/1.1
Accept: application/xml;
detail=subcollection1+subcollection2+subcollection3
```

The API supports extended sub-collections for the following main collections.

Table 7.4. Collections that use extended sub-collections

Collection	Extended Sub-Collection Support
hosts	**statistics**
vms	**statistics**, **nics**, **disks**

Example 7.1. A request for extended statistics, NICs and disks sub-collections in the vms collection

```
GET /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml; detail=statistics+nics+disks
```

7.3.4. Searching Collections with Queries

A **GET** request on a **"collection/search"** link results in a search query of that collection. The API only returns resources within the collection that satisfy the search query constraints.

```
GET /ovirt-engine/api/collection?search={query} HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<collection>
    <resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id">
        ...
```

```
    </resource>
    ...
</collection>
```

7.3.5. Maximum Results Parameter

Use the **max** URL parameter to limit the list of results. An API search query without specifying the **max** parameter will return all values. Specifying the **max** parameter is recommended to prevent API search queries from slowing UI performance.

```
GET /ovirt-engine/api/collection;max=1 HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<collection>
    <resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id">
        <name>Resource-Name</name>
        <description>A description of the resource</description>
        ...
    </resource>
</collection>
```

7.3.6. Case Sensitivity

All search queries are case sensitive by default. The URL syntax provides a Boolean option to toggle case sensitivity.

Example 7.2. Case insensitive search query

```
GET /ovirt-engine/api/collection;case-sensitive=false?search={query}
HTTP/1.1
Accept: application/xml
```

7.3.7. Query Syntax

The API uses the URI templates to perform a search **query** with a **GET** request:

```
GET /ovirt-engine/api/collection?search={query} HTTP/1.1
Accept: application/xml
```

The **query** template value refers to the search query the API directs to the **collection**. This **query** uses the same format as Red Hat Virtualization Query Language:

```
(criteria) [sortby (element) asc|desc]
```

The **sortby** clause is optional and only needed when ordering results.

Table 7.5. Example search queries

Collection	Criteria	Result
hosts	**vms.status=up**	Displays a list of all hosts running virtual machines that are **up**.
vms	**domain=qa.company.com**	Displays a list of all virtual machines running on the specified domain.
vms	**users.name=mary**	Displays a list of all virtual machines belonging to users with the user name **mary**.
events	**severity>normal sortby time**	Displays the list of all **events** with severity higher than **normal** and sorted by the **time** element values.
events	**severity>normal sortby time desc**	Displays the list of all **events** with severity higher than **normal** and sorted by the **time** element values in descending order.

The API requires the **query** template to be URL-encoded to translate reserved characters, such as operators and spaces.

Example 7.3. URL-encoded search query

```
GET /ovirt-engine/api/vms?search=name%3Dvm1 HTTP/1.1
Accept: application/xml
```

7.3.8. Wildcards

Search queries substitute part of a value with an asterisk as a wildcard.

Example 7.4. Wildcard search query for name=vm*

```
GET /ovirt-engine/api/vms?search=name%3Dvm* HTTP/1.1
Accept: application/xml
```

This query would result in all virtual machines with names beginning with **vm**, such as **vm1**, **vm2**, **vma** or **vm-webserver**.

Example 7.5. Wildcard search query for name=v*1

```
GET /ovirt-engine/api/vms?search=name%3Dv*1 HTTP/1.1
Accept: application/xml
```

This query would result in all virtual machines with names beginning with **v** and ending with **1**, such as **vm1**, **vr1** or **virtualmachine1**.

7.3.9. Pagination

Some Red Hat Virtualization environments contain large collections of resources. However, the API only displays a default number of resources for one search query to a collection. To display more than the default, the API separates collections into pages via a search query containing the **page** command.

Example 7.6. Paginating resources

This example paginates resources in a collection. The URL-encoded request is:

```
GET /ovirt-engine/api/collection?search=page%201 HTTP/1.1
Accept: application/xml
```

Increase the **page** value to view the next page of results:

```
GET /ovirt-engine/api/collection?search=page%202 HTTP/1.1
Accept: application/xml
```

Use the **page** command in conjunction with other commands in a search query. For example:

```
GET /ovirt-engine/api/collection?
search=sortby%20element%20asc%20page%202 HTTP/1.1
Accept: application/xml
```

This query displays the second page in a collection listing ordered by a chosen element.

IMPORTANT

The REST APIs are stateless; it is not possible to retain a state between different requests since all requests are independent from each other. As a result, if a status change occurs between your requests, then the page results may be inconsistent.

For example, if you request a specific page from a list of VMs, and a status change occurs before you can request the next page, then your results may be missing entries or contain duplicated entries.

7.3.10. Creating a Resource in a Collection

Create a new resource with a **POST** request to the collection URI containing a representation of the new resource.

A **POST** request requires a **Content-Type** header. This informs the API of the representation MIME type in the body content as part of the request.

Include an **Accept** HTTP header to define the MIME type for the response format.

Each resource type has its own specific required properties. The client supplies these properties when creating a new resource. Refer to the individual resource type documentation for more details.

If a required property is absent, the creation fails with a representation indicating the missing elements.

```
POST /ovirt-engine/api/[collection] HTTP/1.1
Accept: [MIME type]
Content-Type: [MIME type]

[body]
```

7.3.11. Asynchronous Requests

The API performs asynchronous **POST** requests unless the user overrides them with an **Expect: 201-created** header.

For example, certain resources, such as Virtual Machines, Disks, Snapshots and Templates, are created asynchronously. A request to create an asynchronous resource results in a **202 Accepted** status. The initial document structure for a **202 Accepted** resource also contains a **creation_status** element and link for creation status updates. For example:

```
POST /ovirt-engine/api/collection HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<resource>
    <name>Resource-Name</name>
</resource>

HTTP/1.1 202 Accepted
Content-Type: application/xml

<resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id">
    <name>Resource-Name</name>
    <creation_status>
        <state>pending</state>
    </creation status>
    <link rel="creation_status"
      href="/ovirt-
engine/api/collection/resource_id/creation_status/creation_status_id"/>
      ...
</resource>
```

A **GET** request to the **creation_status** link provides a creation status update:

```
GET /ovirt-
engine/api/collection/resource_id/creation_status/creation_status_id
HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<creation id="creation_status_id"
  href="/ovirt-
engine/api/collection/resource_id/creation_status/creation_status_id">
```

```
    <status>
        <state>complete</state>
    </status>
</creation>
```

Overriding the asynchronous resource creation requires an **`Expect: 201-created`** header:

```
POST /ovirt-engine/api/collection HTTP/1.1
Accept: application/xml
Content-Type: application/xml
Expect: 201-created

<resource>
    <name>Resource-Name</name>
</resource>
```

7.4. RESOURCES

7.4.1. Resources

Resources are data sources in a RESTful web service. Each resource type contains a set of common parameters that the REST API abstracts to form a **resource representation**, usually in XML or JSON. Users can view a resource representation, then edit the parameters and send the representation back to the resource's URL within the API, which modifies the resource. Users can also delete individual resources through REST.

A RESTful web service also groups resources into **collections**. Users can view a representation of all resources in a collection. Users also send resource representations to a specific collection to create a new resource within that particular collection.

7.4.2. Retrieving a Resource

Obtain the state of a resource with a **`GET`** request on a URI obtained from a collection listing.

Include an **`Accept`** HTTP header to define the MIME type for the response format.

```
GET /ovirt-engine/api/[collection]/[resource_id] HTTP/1.1
Accept: [MIME type]
```

You can obtain additional information from some resources using the **`All-Content: true`** header. The RESTful Service Description Language describes which links support this header.

```
GET /ovirt-engine/api/[collection]/[resource_id] HTTP/1.1
Accept: [MIME type]
All-Content: true
```

7.4.3. Updating a Resource

Modify resource properties with a **`PUT`** request containing an updated description from a previous **`GET`** request for the resource URI. Details on modifiable properties are found in the individual resource type documentation.

A **PUT** request requires a **Content-Type** header. This informs the API of the representation MIME type in the body content as part of the request.

Include an **Accept** HTTP header to define the MIME type for the response format.

```
PUT /ovirt-engine/api/collection/resource_id HTTP/1.1
Accept: [MIME type]
Content-Type: [MIME type]

[body]
```

This does not include immutable resource properties that an API user has attempted to modify. If an attempt is made to modify a *strictly* immutable resource property, the API reports a conflict with an error message representation in the response body.

Properties omitted from the representation are ignored and not changed.

7.4.4. Deleting a Resource

Delete a resource with a **DELETE** request sent to its URI.

Include an **Accept** HTTP header to define the MIME type for the response format.

```
DELETE /ovirt-engine/api/[collection]/[resource_id] HTTP/1.1
Accept: [MIME type]
```

Some cases require optional body content in the **DELETE** request to specify additional properties. A **DELETE** request with optional body content requires a **Content-Type** header to inform the API of the representation MIME type in the body content. If a **DELETE** request contains no body content, omit the **Content-Type** header.

7.4.5. Sub-Collection Relationships

A sub-collection relationship defines a hierarchical link between a resource and a sub-collection. The sub-collection exists or has some meaning in the context of a parent resource. For example, a virtual machine contains network interfaces, which means the API maps the relationship between the virtual machine resource and the network interfaces sub-collection.

Sub-collections are used to model the following relationships types:

- Where one parent resource can contain several child resources and vice versa. For example, a virtual machine can contain several disks and some disks are shared among multiple virtual machines.
- Where mapped resources are dependent on a parent resource. Without the parent resource, the dependent resource cannot exist. For example, the link between a virtual machine and snapshots.
- Where mapped resources exist independently from parent resources but data is still associated with the relationship. For example, the link between a cluster and a network.

The API defines a relationship between a resource and a sub-collection using the **link rel=** attribute:

```
GET /ovirt-engine/api/collection/resource_id HTTP/1.1
```

```
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id">
    ...
    <link rel="subcollection"
      href="/ovirt-engine/api/collection/resource_id/subcollection"/>
    ...
</resource>
```

The API user now queries the sub-collection.

```
GET /ovirt-engine/api/collection/resource_id/subcollection HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<subcollection>
    <subresource id="subresource_id"
      href="/ovirt-
engine/api/collection/resource_id/subcollection/subresource_id">
        ...
    </subresource>
    ...
</subcollection>
```

7.4.6. XML Element Relationships

XML element links act as an alternative to sub-collections to express relationships between resources. XML element links are simply elements with a "href" attribute that points to the linked element.

XML element links are used to model simple 1:N mappings between resources without a dependency and without data associated with the relationship. For example, the relationship between a host and a cluster.

Examples of such relationships include:

- Backlinks from a resource in a sub-collection to a parent resource; or
- Links between resources with an arbitrary relationship.

Example 7.7. Backlinking from a sub-collection resource to a resource using an XML element

```
GET /ovirt-
engine/api/collection/resource_id/subcollection/subresource_id HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/xml

<subcollection>
    <subresource id="subresource_id"
```

```
      href="/ovirt-
engine/api/collection/resource_id/subcollection/subresource_id">
        <resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id"/>
        ...
    </subresource>
</subcollection>
```

7.4.7. Actions

Most resources include a list of action links to provide functions not achieved through the standard HTTP methods.

```
<resource>
    ...
    <actions>
        <link rel="start" href="/ovirt-
engine/api/collection/resource_id/start"/>
        <link rel="stop" href="/ovirt-
engine/api/collection/resource_id/stop"/>
        ...
    </actions>
    ...
</resource>
```

The API invokes an action with a **POST** request to the supplied URI. The body of the **POST** requires an **action** representation encapsulating common and task-specific parameters.

Table 7.6. Common action parameters

Element	Description
async	**true** if the server responds immediately with **202 Accepted** and an action representation contains a **href** link to be polled for completion.
grace_period	a grace period in milliseconds, which must expire before the action is initiated.

Individual actions and their parameters are documented in the individual resource type's documentation. Some parameters are mandatory for specific actions and their absence is indicated with a **fault** response.

An action also requires a **Content-Type: application/xml** header since the **POST** request requires an XML representation in the body content.

When the action is initiated asynchronously, the immediate **202 Accepted** response provides a link to monitor the status of the task:

```
POST /ovirt-engine/api/collection/resource_id/action HTTP/1.1
Content-Type: application/xml
Accept: application/xml
```

```
<action>
    <async>true</async>
</action>

HTTP/1.1 202 Accepted
Content-Type: application/xml

<action id="action_id"
  href="/ovirt-engine/api/collection/resource_id/action/action_id">
    <async>true</async>
    ...
</action>
```

A subsequent **GET** on the action URI provides an indication of the status of the asynchronous task.

Table 7.7. Action statuses

Status	Description
pending	Task has not yet started.
in_progress	Task is in operation.
complete	Task completed successfully.
failed	Task failed. The returned **action** representation would contain a **fault** describing the failure.

Once the task has completed, the action is retained for an indeterminate period. Once this has expired, subsequent **GET**s are **301 Moved Permanently** redirected back to the target resource.

```
GET /ovirt-engine/api/collection/resource_id/action/action_id HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<action id="action_id"
  href="/ovirt-engine/api/collection/resource_id/action/action_id">
    <status>
        <state>pending</state>
    </status>
    <link rel="parent" /ovirt-engine/api/collection/resource_id"/>
    <link rel="replay" href="/ovirt-
engine/api/collection/resource_id/action"/>
</action>
```

An action representation also includes some links that are identified by the **rel** attribute:

Table 7.8. Action relationships

Type	Description
parent	A link back to the resource of this action.
replay	A link back to the original action URI. POSTing to this URI causes the action to be re-initiated.

7.4.8. Permissions

Each resource contains a **permissions** sub-collection. Each **permission** contains a **user**, an assigned **role** and the specified resource. For example:

```
GET /ovirt-engine/api/collection/resource_id/permissions HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<permissions>
    <permission id="permission-id"
      href="/ovirt-
engine/api/collection/resource_id/permissions/permission_id">
        <role id="role_id" href="/ovirt-engine/api/roles/role_id"/>
        <user id="user_id" href="/ovirt-engine/api/users/user_id"/>
        <resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id"/>
    </permission>
    ...
</permissions>
```

A resource acquires a new permission when an API user sends a **POST** request with a **permission** representation and a **Content-Type: application/xml** header to the resource's **permissions** sub-collection. Each new permission requires a **role** and a **user**:

```
POST /ovirt-engine/api/collection/resource_id/permissions HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<permission>
    <role id="role_id"/>
    <user id="user_id"/>
</permission>

HTTP/1.1 201 Created
Content-Type: application/xml

<permission id="permission_id"
  href="/ovirt-
engine/api/resources/resource_id/permissions/permission_id">
    <role id="role_id" href="/ovirt-engine/api/roles/role_id"/>
    <user id="user_id" href="/ovirt-engine/api/users/user_id"/>
```

```
    <resource id="resource_id" href="/ovirt-
engine/api/collection/resource_id"/>
</permission>
```

7.4.9. Handling Errors

Some errors require further explanation beyond a standard HTTP status code. For example, the API reports an unsuccessful resource state update or action with a **`fault`** representation in the response entity body. The fault contains a **`reason`** and **`detail`** strings. Clients must accommodate failed requests via extracting the **`fault`** *or* the expected resource representation depending on the response status code. Such cases are clearly indicated in the individual resource documentation.

```
PUT /ovirt-engine/api/collection/resource_id HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<resource>
    <id>id-update-test</id>
</resource>

HTTP/1.1 409 Conflict
Content-Type: application/xml

<fault>
    <reason>Broken immutability constraint</reason>
    <detail>Attempt to set immutable field: id</detail>
</fault>
```

CHAPTER 8. THE BACKUP AND RESTORE API

The backup and restore API is a collection of functions that allows you to perform full or file-level backup and restoration of virtual machines. The API combines several components of Red Hat Virtualization, such as live snapshots and the REST API, to create and work with temporary volumes that can be attached to a virtual machine containing backup software provided by an independent software provider.

For supported third-party backup vendors, consult the Red Hat Virtualization Ecosystem at Red Hat Marketplace.

8.1. BACKING UP A VIRTUAL MACHINE

Use the backup and restore API to back up a virtual machine. This procedure assumes you have two virtual machines: the virtual machine to back up, and a virtual machine on which the software for managing the backup is installed.

Procedure 8.1. Backing Up a Virtual Machine

1. Using the REST API, create a snapshot of the virtual machine to back up:

```
POST /ovirt-engine/api/vms/11111111-1111-1111-1111-
111111111111/snapshots/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<snapshot>
    <description>BACKUP</description>
</snapshot>
```

NOTE

When you take a snapshot of a virtual machine, a copy of the configuration data of the virtual machine as at the time the snapshot was taken is stored in the **data** attribute of the **configuration** attribute in **initialization** under the snapshot.

IMPORTANT

You cannot take snapshots of disks that are marked as shareable or that are based on direct LUN disks.

2. Retrieve the configuration data of the virtual machine from the **data** attribute under the snapshot:

```
GET /ovirt-engine/api/vms/11111111-1111-1111-1111-
111111111111/snapshots/11111111-1111-1111-1111-111111111111 HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

3. Identify the disk ID and snapshot ID of the snapshot:

```
GET /ovirt-engine/api/vms/11111111-1111-1111-1111-
```

```
11111111111/snapshots/11111111-1111-1111-1111-11111111111/disks
HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

4. Attach the snapshot to the backup virtual machine and activate the disk:

```
POST /ovirt-engine/api/vms/22222222-2222-2222-2222-
222222222222/disks/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk id="11111111-1111-1111-1111-11111111111">
    <snapshot id="11111111-1111-1111-1111-11111111111"/>
    <active>true</active>
</disk>
```

5. Use the backup software on the backup virtual machine to back up the data on the snapshot disk.

6. Detach the snapshot disk from the backup virtual machine:

```
DELETE /ovirt-engine/api/vms/22222222-2222-2222-2222-
222222222222/disks/11111111-1111-1111-1111-11111111111 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <detach>true</detach>
</action>
```

7. Optionally, delete the snapshot:

```
DELETE /ovirt-engine/api/vms/11111111-1111-1111-1111-
11111111111/snapshots/11111111-1111-1111-1111-11111111111 HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

You have backed up the state of a virtual machine at a fixed point in time using backup software installed on a separate virtual machine.

8.2. RESTORING A VIRTUAL MACHINE

Restore a virtual machine that has been backed up using the backup and restore API. This procedure assumes you have a backup virtual machine on which the software used to manage the previous backup is installed.

Procedure 8.2. Restoring a Virtual Machine

1. Attach the disk to the backup virtual machine:

```
POST /ovirt-engine/api/vms/22222222-2222-2222-2222-
222222222222/disks/ HTTP/1.1
```

```
Accept: application/xml
Content-type: application/xml

<disk id="11111111-1111-1111-1111-111111111111">
</disk>
```

2. Use the backup software to restore the backup to the disk.

3. Detach the disk from the backup virtual machine:

```
DELETE /ovirt-engine/api/vms/22222222-2222-2222-2222-
222222222222/disks/11111111-1111-1111-1111-111111111111 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <detach>true</detach>
</action>
```

4. Create a new virtual machine using the configuration data of the virtual machine being restored:

```
POST /ovirt-engine/api/vms/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <cluster>
        <name>cluster_name</name>
    </cluster>
    <name>NAME</name>
    ...
</vm>
```

5. Attach the disk to the new virtual machine:

```
POST /ovirt-engine/api/vms/33333333-3333-3333-3333-
333333333333/disks/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk id="11111111-1111-1111-1111-111111111111">
</disk>
```

You have restored a virtual machine using a backup that was created using the backup and restore API.

CHAPTER 9. DATA CENTERS

9.1. DATA CENTER ELEMENTS

The **datacenters** collection provides information about the data centers in a Red Hat Virtualization environment. An API user accesses this information through the **rel="datacenters"** link obtained from the entry point URI.

The following table shows specific elements contained in a data center resource representation.

Table 9.1. Data center elements

Element	Type	Description	Properties
name	string	A plain text, human-readable name for the data center. The **name** is unique across all data center resources.	
description	string	A plain text, human-readable description of the data center	
link rel="storagedomains"	relationship	A link to the sub-collection for storage domains attached to this data center.	
link rel="clusters"	relationship	A link to the sub-collection for clusters attached to this data center.	
link rel="networks"	relationship	A link to the sub-collection for networks available to this data center.	
link rel="permissions"	relationship	A link to the sub-collection for data center permissions.	
link rel="quotas"	relationship	A link to the sub-collection for quotas associated with this data center.	
local	Boolean: true or false	Specifies whether the data center is a local data center, such as created in all-in-one instances.	
storage_format	enumerated	Describes the storage format version for the data center. A list of enumerated values are available in **capabilities**.	

Element	Type	Description	Properties
version major= minor=	complex	The compatibility level of the data center.	
supported_versions	complex	A list of possible version levels for the data center, including **version major= minor=**.	
mac_pool	string	The MAC address pool associated with the data center. If no MAC address pool is specified the default MAC address pool is used.	
status	see below	The data center status.	

The **status** contains one of the following enumerated values: **uninitialized**, **up**, **maintenance**, **not_operational**, **problematic** and **contend**. These states are listed in **data_center_states** under **capabilities**.

9.2. XML REPRESENTATION OF A DATA CENTER

Example 9.1. An XML representation of a data center

```
<data_center href="/ovirt-engine/api/datacenters/00000000-0000-0000-
0000-000000000000"
    id="00000000-0000-0000-0000-000000000000">
  <name>Default</name>
  <description>The default Data Center</description>
  <link href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/storagedomains" rel="storagedomains"/>
  <link href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/clusters" rel="clusters"/>
  <link href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/networks" rel="networks"/>
  <link href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/permissions" rel="permissions"/>
  <link href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/quotas" rel="quotas"/>
  <local>false</local>
  <storage_format>v3</storage_format>
  <version major="4" minor="0"/>
  <supported_versions>
    <version major="4" minor="0"/>
  </supported_versions>
  <status>
    <state>up</state>
  </status>
```

```
  <mac_pool href="/ovirt-engine/api/macpools/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000"/>
</data_center>
```

9.3. JSON REPRESENTATION OF A DATA CENTER

Example 9.2. A JSON representation of a data center

```
{
  "data_center" : [ {
    "local" : "false",
    "storage_format" : "v3",
    "version" : {
      "major" : "4",
      "minor" : "0"
    },
    "supported_versions" : {
      "version" : [ {
        "major" : "4",
        "minor" : "0"
      } ]
    },
    "status" : {
      "state" : "up"
    },
    "mac_pool":
            {
                "href": "/ovirt-engine/api/macpools/00000000-0000-0000-
0000-000000000000",
                "id": "00000000-0000-0000-0000-000000000000"
            },
    "name" : "Default",
    "description" : "The default Data Center",
    "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255",
    "id" : "00000002-0002-0002-0002-000000000255",
    "link" : [ {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/storagedomains",
      "rel" : "storagedomains"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/clusters",
      "rel" : "clusters"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/networks",
      "rel" : "networks"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/permissions",
      "rel" : "permissions"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
```

```
000000000255/quotas",
      "rel" : "quotas"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/iscsibonds",
      "rel" : "iscsibonds"
    }, {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255/qoss",
      "rel" : "qoss"
    } ]
  } ]
}
```

9.4. METHODS

9.4.1. Creating a New Data Center

Creation of a new data center requires the **name** and **local** elements.

Example 9.3. Creating a data center

```
POST /ovirt-engine/api/datacenters HTTP/1.1
Accept: application/xml
Content-type: application/xml

<data_center>
    <name>NewDatacenter</name>
    <local>false</local>
</data_center>
```

9.4.2. Updating a Data Center

The **name**, **description**, **storage_type**, **version**, **storage_format** and **mac_pool** elements are updatable post-creation.

Example 9.4. Updating a data center

```
PUT /ovirt-engine/api/datacenters/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<data_center>
    <name>UpdatedName</name>
    <description>An updated description for the data
center</description>
</data_center>
```

9.4.3. Removing a Data Center

Removal of a data center requires a **DELETE** request.

Example 9.5. Removing a data center

```
DELETE /ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000 HTTP/1.1

HTTP/1.1 204 No Content
```

9.5. SUB-COLLECTIONS

9.5.1. Storage Domains Sub-Collection

9.5.1.1. Storage Domains Sub-Collection

Each data center contains a sub-collection for attached storages domain. An API user interacts with this sub-collection using the standard REST methods.

An attached storage domain has a similar representation to a top-level storage domain, with the exception that it has a data center specific **status** and set of actions. States for the **status** element are listed in **storage_domain_states** under **capabilities**.

IMPORTANT

The API as documented in this section is experimental and subject to change. It is not covered by the backwards compatibility statement.

9.5.1.2. Attaching and Detaching a Storage Domain

A data center is only ready for use when at least one storage domain is attached, which an API user **POST**s to the data center's storage domains sub-collection.

When attaching a storage domain, its **id** or **name** must be supplied. An example of attaching a storage domain to a data center:

Example 9.6. Attach a storage domain to a data center

```
POST /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"/>

HTTP/1.1 201 Created
Location: /datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/fabe0451-701f-4235-8f7e-e20e458819ed
Content-Type: application/xml
```

```
<storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"
  href="/ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/
  fabe0451-701f-4235-8f7e-e20e458819ed">
    <name>images0</name>
    <type>data</type>
    <status>
        <state>inactive</state>
    </status>
    <master>true</master>
    <storage>
        <type>nfs</type>
        <address>172.31.0.6</address>
        <path>/exports/RHEVX/images/0</path>
    </storage>
    <data_center id="d70d5e2d-b8ad-494a-a4d2-c7a5631073c4"
      href="/ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4"/>
    <actions>
        <link rel="activate"
          href="/ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/
          storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/activate"/>
        <link rel="deactivate"
          href="/ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/
          storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/deactivate"/>
    </actions>
</storage_domain>
```

Detach a storage domain from a data center with a **DELETE** request. Include an optional **async** element for this request to be asynchronous.

Example 9.7. Detach a storage domain from a data center

```
DELETE /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/fabe0451-701f-4235-8f7e-e20e458819ed
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
  <async>true</async>
</action>
```

9.5.1.3. Actions

9.5.1.3.1. Activate Storage Domain Action

An attached storage domain requires activation on a data center before use. The activate action does not take any action specific parameters.

Example 9.8. Action to active a storage domain on a datacenter

```
POST /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/activate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

9.5.1.3.2. Deactivate Storage Domain Action

An attached storage domain is deactivated on a data center before removal. The deactivate action does not take any action specific parameters.

Example 9.9. Action to deactivate a storage domain on a datacenter

```
POST /ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/deactivate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

9.5.2. Network Sub-Collection

9.5.2.1. Networks Sub-Collection

Networks associated with a data center are represented with the **`networks`** sub-collection. The representation of a data center's **`network`** sub-collection contains the following elements:

Table 9.2. Network elements

Element	Type	Description
`name`	string	A plain text, human readable name for the network.
`description`	string	A plain text, human readable description of the network.
`rel="permissions"`	relationship	A link to the **`permissions`** sub-collection for the network.

Element	Type	Description
rel="vnicprofiles"	relationship	A link to the **vnicprofiles** sub-collection for the network.
rel="labels"	relationship	A link to the **labels** sub-collection for the network.
data_center id=	relationship	A reference to the data center of which the network is a member.
stp	Boolean: true or false	Specifies whether spanning tree protocol is enabled for the network.
mtu	integer	Specifies the maximum transmission unit for the network.
usages	complex	Defines a set of **usage** elements for the network. Users can define networks as **vm** and **display** networks at this level.

In the REST API, you can manipulate the **networks** sub-collection with the standard REST methods. For example, the **POST** method can be used to update a network **id** or **name**

Example 9.10. Associating a network resource with a data center

```
POST /ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/networks HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network id="da05ac09-00be-45a1-b0b5-4a6a2438665f">
    <name>ovirtmgmt</name>
</network>

HTTP/1.1 201 Created
Location: http://{host}/clusters/00000000-0000-0000-0000-
000000000000/networks/00000000-0000-0000-0000-000000000000
Content-Type: application/xml

<network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000"
    id="00000000-0000-0000-0000-000000000000">
  <name>Network_001</name>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/permissions"
    rel="permissions"/>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/vnicprofiles"
    rel="vnicprofiles"/>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/labels"
```

```
    rel="labels"/>
  <data_center href="/ovirt-engine/api/datacenters/00000000-0000-0000-
0000-000000000000"
    id="00000000-0000-0000-0000-000000000000"/>
  <stp>false</stp>
  <mtu>0</mtu>
  <usages>
    <usage>vm</usage>
  </usages>
</network>
```

Update the resource with a **PUT** request. The maximum transmission unit of a network is set using a **PUT** request to specify the integer value of the **mtu** element.

Example 9.11. Setting the network maximum transmission unit

```
PUT /ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/networks/00000000-0000-0000-0000-000000000000 HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network>
    <mtu>1500</mtu>
</network>
```

An association is removed with a **DELETE** request to the appropriate element in the collection.

Example 9.12. Removing a network association from a data center

```
DELETE /ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000/networks/00000000-0000-0000-0000-000000000000 HTTP/1.1

HTTP/1.1 204 No Content
```

9.5.3. Quotas Sub-Collection

9.5.3.1. Quotas Sub-Collection

The quotas sub-collection lists restrictions that Red Hat Virtualization Manager implements on resources. An API user views this sub-collection and its resources using the **GET** method.

Example 9.13. An XML representation of a quota

```
<quota href="/ovirt-engine/api/datacenters/56087282-d7a6-11e1-af44-
001a4a400e0c
  /quotas/e13ff85a-b2ba-4f7b-8010-e0d057c03dfe"
  id="e13ff85a-b2ba-4f7b-8010-e0d057c03dfe">
    <name>MyQuota</name>
    <description>A quota for my Red Hat Enterprise
```

```
      Virtualization environment</description>
    <data_center href= "/ovirt-engine/api/datacenters/56087282-d7a6-
11e1-af44-001a4a400e0c"
    id="56087282-d7a6-11e1-af44-001a4a400e0c"/>
</quota>
```

Creation of a new quota requires the **name** and **description** elements.

Example 9.14. Creating a quota

```
POST /ovirt-engine/api/datacenters/56087282-d7a6-11e1-af44-
001a4a400e0c/quotas HTTP/1.1
Accept: application/xml
Content-type: application/xml

<quota>
    <name>VMQuota</name>
    <description>My new quota for virtual machines</description>
</quota>
```

Removal of a quota requires a **DELETE** request.

Example 9.15. Removing a quota

```
DELETE /ovirt-engine/api/datacenters/01a45ff0-915a-11e0-8b87-
5254004ac988/quotas/e13ff85a-b2ba-4f7b-8010-e0d057c03dfe HTTP/1.1

HTTP/1.1 204 No Content
```

9.6. ACTIONS

9.6.1. Force Remove Data Center Action

An API user forces the removal of a data center when encountering unresolvable problems with storage domains, such as the loss of connection to a master storage domain or a lack of available hosts when deleting storage domains. The API includes a **force** action to help with these situations.

This action removes database entries associated with a chosen data center before the API removes the data center from the Red Hat Virtualization environment. This means the API removes the data center regardless of associated storage domains.

This action requires a **DELETE** method. The request body contains an **action** representation with the **force** parameter set to **true**. The request also requires an additional **Content-type: application/xml** header to process the XML representation in the body.

Example 9.16. Force remove action on a data center

```
DELETE /ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000 HTTP/1.1
```

```
Accept: application/xml
Content-type: application/xml

<action>
  <force>true</force>
</action>
```

This action:

- Deletes all database information for **data** storage domains associated the data center;
- Deletes all database information for resources, such as virtual machines and templates, on **data** storage domains associated the data center;
- Detaches **iso** and **export** storage domains from the data center; and
- Deletes the database information for the data center.

This action overrides the requirement for a data center to be empty before deletion.

IMPORTANT

This action only removes the database entries for resources associated with the data center. The **data** storage domains associated with the data center require manual format before reuse. Metadata for **iso** and **export** domains require manual cleaning prior to use on another data center.

CHAPTER 10. CLUSTERS

10.1. CLUSTER ELEMENTS

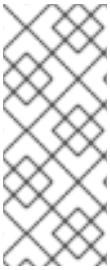
The **clusters** collection provides information about clusters in a Red Hat Virtualization environment. An API user accesses this information through the **rel="clusters"** link obtained from the entry point URI.

The following table shows specific elements contained in a cluster resource representation.

Table 10.1. Cluster elements

Element	Type	Description	Properties
name	string	A user-supplied, human-readable name for the cluster. The **name** is unique across all cluster resources.	
description	string	A free-form, user-supplied, human-readable description of the cluster.	
link rel="networks"	relationship	A link to the sub-collection for networks associated with this cluster.	
link rel="permissions"	relationship	A link to the sub-collection for cluster permissions.	
link rel="glustervolumes"	relationship	A link to the sub-collection for Red Hat Gluster Storage volumes associated with this cluster.	
link rel="glusterhooks"	relationship	A link to the sub-collection for Red Hat Gluster Storage volume hooks associated with this cluster.	
link rel="affinitygroups"	relationship	A link to the sub-collection for virtual machine affinity groups associated with this cluster.	
cpu id=	complex	A server CPU reference that defines the CPU type all hosts must support in the cluster.	
data_center id=	GUID	A reference to the data center membership of this cluster.	 

Element	Type	Description	Properties
memory_policy	complex	Defines the cluster's policy on host memory utilization.	
scheduling_policy	complex	Defines the load-balancing or power-saving modes for hosts in the cluster.	
version major= minor=	complex	The compatibility level of the cluster.	
supported_versions	complex	A list of possible **version** levels for the cluster.	
error_handling	complex/enumerated	Defines virtual machine handling when a host within a cluster becomes non-operational. Requires a single **on_error** element containing an enumerated type property listed in **capabilities**.	
virt_service	Boolean	Defines whether to expose virtualization services for this cluster.	
gluster_service	Boolean	Defines whether to expose Red Hat Gluster Storage services for this cluster.	
threads_as_cores	Boolean	Defines whether hosts can run virtual machines with a total number of processor cores greater than the number of cores in the host.	
tunnel_migration	Boolean	Defines whether virtual machines use a libvirt-to-libvirt tunnel during migration.	
trusted_service	Boolean	Defines whether an OpenAttestation server is used to verify hosts.	
ballooning_enabled	Boolean	Defines whether ballooning is enabled for the cluster.	
ksm	Boolean	Defines whether ksm is enabled for the cluster.	



NOTE

When a host's free memory drops below 20%, ballooning commands like **`mom.Controllers.Balloon - INFO Ballooning guest:half1 from 1096400 to 1991580`** are logged to **`/etc/vdsm/mom.conf`**. **`/etc/vdsm/mom.conf`** is the Memory Overcommit Manager log file. An event will also be added to the event log if a virtual machine does not respect a balloon.

10.2. MEMORY POLICY ELEMENTS

The **`memory_policy`** element contains the following elements:

Table 10.2. Memory policy elements

Element	Type	Description	Properties
`overcommit percent=`	complex	The percentage of host memory allowed in use before no more virtual machines can start on a host. Virtual machines can use more than the available host memory due to memory sharing under KSM. Recommended values include **`100`** (None), **`150`** (Server Load) and **`200`** (Desktop Load).	
`transparent_hugepages`	complex	Define the **`enabled`** status of Transparent Hugepages. The status is either true or false. Check **`capabilities`** feature set to ensure your version supports **`transparent hugepages`**.	

10.3. SCHEDULING POLICY ELEMENTS

The **`scheduling_policy`** element contains the following elements:

Table 10.3. Scheduling policy elements

Element	Type	Description	Properties
`policy`	enumerated	The VM scheduling mode for hosts in the cluster. A list of enumerated types are listed in **`capabilities`**.	

Element	Type	Description	Properties
`thresholds low= high= duration=`	complex	Defines CPU limits for the host. The **`high`** attribute controls the highest CPU usage percentage the host can have before being considered overloaded. The **`low`** attribute controls the lowest CPU usage percentage the host can have before being considered underutilized. The **`duration`** attribute refers to the number of seconds the host needs to be overloaded before the scheduler starts and moves the load to another host.	

10.4. XML REPRESENTATION OF A CLUSTER

Example 10.1. An XML representation of a cluster

```
<cluster id="00000000-0000-0000-0000-000000000000"
  href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000">
    <name>Default</name>
    <description>The default server cluster</description>
    <link rel="networks"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000/networks"/>
    <link rel="permissions"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000/permissions"/>
          <link rel="glustervolumes"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000/glustervolumes"/>
          <link rel="glusterhooks"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000/glusterhooks"/>
          <link rel="affinitygroups"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000/affinitygroups"/>
    <cpu id="Intel Penryn Family"/>
        <architecture>X86_64<architecture/>
    <data_center id="00000000-0000-0000-0000-000000000000"
      href="/ovirt-engine/api/datacenters/00000000-0000-0000-0000-
000000000000"/>
    <memory_policy>
        <overcommit percent="100"/>
        <transparent_hugepages>
            <enabled>false</enabled>
        </transparent_hugepages>
    </memory_policy>
    <scheduling_policies>
      <policy>evenly_distributed</policy>
      <thresholds low="10" high="75" duration="120"/>
```

```
    </scheduling_policies>
    <version major="4" minor="0"/>
    <supported_versions>
        <version major="4" minor="0"/>
    </supported_versions>
    <error_handling>
        <on_error>migrate</on_error>
    </error_handling>
    <virt_service>true</virt_service>
    <gluster_service>false</gluster_service>
    <threads_as_cores>false</threads_as_cores>
    <tunnel_migration>false</tunnel_migration>
    <trusted_service>false</trusted_service>
    <ha_reservation>false</ha_reservation>
    <ballooning_enabled>false</ballooning_enabled>
    <ksm>
        <enabled>true</enabled>
    </ksm>
</cluster>
```

10.5. JSON REPRESENTATION OF A CLUSTER

Example 10.2. A JSON representation of a cluster

```
{
  "cluster" : [ {
    "cpu" : {
      "architecture" : "X86_64",
      "id" : "Intel Penryn Family"
    },
    "data_center" : {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255",
      "id" : "00000002-0002-0002-0002-000000000255"
    },
    "memory_policy" : {
      "overcommit" : {
        "percent" : "100"
      },
      "transparent_hugepages" : {
        "enabled" : "true"
      }
    },
    "scheduling_policy" : {
      "policy" : "none",
      "name" : "none",
      "href" : "/ovirt-engine/api/schedulingpolicies/b4ed2332-a7ac-4d5f-
9596-99a439cb2812",
      "id" : "b4ed2332-a7ac-4d5f-9596-99a439cb2812"
    },
    "version" : {
      "major" : "4",
      "minor" : "0"
    },
```

```
    "error_handling" : {
      "on_error" : "migrate"
    },
    "virt_service" : "true",
    "gluster_service" : "false",
    "threads_as_cores" : "false",
    "tunnel_migration" : "false",
    "trusted_service" : "false",
    "ha_reservation" : "false",
    "optional_reason" : "false",
    "ballooning_enabled" : "false",
    "ksm" : {
      "enabled" : "true"
    },
    "required_rng_sources" : { },
    "name" : "Default",
    "description" : "The default server cluster",
    "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb",
    "id" : "00000001-0001-0001-0001-0000000002fb",
    "link" : [ {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/networks",
      "rel" : "networks"
    }, {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/permissions",
      "rel" : "permissions"
    }, {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/glustervolumes",
      "rel" : "glustervolumes"
    }, {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/glusterhooks",
      "rel" : "glusterhooks"
    }, {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/affinitygroups",
      "rel" : "affinitygroups"
    }, {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb/cpuprofiles",
      "rel" : "cpuprofiles"
    } ]
  } ]
}
```


10.6. METHODS

10.6.1. Creating a Cluster

Creation of a new cluster requires the **name**, **cpu id=** and **datacenter** elements. Identify the **datacenter** with either the **id** attribute or **name** element.

Example 10.3. Creating a cluster

```
POST /ovirt-engine/api/clusters HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cluster>
    <name>cluster1</name>
    <cpu id="Intel Penryn Family"/>
    <data_center id="00000000-0000-0000-0000-000000000000"/>
</cluster>
```

10.6.2. Updating a Cluster

The **name**, **description**, **cpu id=** and **error_handling** elements are updatable post-creation.

Example 10.4. Updating a cluster

```
PUT /ovirt-engine/api/clusters/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cluster>
    <description>Cluster 1</description>
</cluster>
```

10.6.3. Removing a Cluster

Removal of a cluster requires a **DELETE** request.

Example 10.5. Removing a cluster

```
DELETE /ovirt-engine/api/clusters/00000000-0000-0000-0000-000000000000
HTTP/1.1

HTTP/1.1 204 No Content
```

10.7. SUB-COLLECTIONS

10.7.1. Networks Sub-Collection

10.7.1.1. Networks Sub-Collection

Networks associated with a cluster are represented with the **networks** sub-collection. Every host within a cluster connects to these associated networks.

The representation of a cluster's **network** sub-collection is the same as a standard **network** resource except for the following additional elements:

Table 10.4. Additional network elements

Element	Type	Description	Properties
cluster id=	relationship	A reference to the cluster of which this network is a member.	
required	Boolean	Defines required or optional network status.	
display	Boolean	Defines the display network status. Used for backward compatibility.	
usages	complex	Defines a set of **usage** elements for the network. Users can define networks as **VM** and **DISPLAY** networks at this level.	

An API user manipulates the **networks** sub-collection with the standard REST methods. **POST**ing a network **id** or **name** reference to the **networks** sub-collection associates the network with the cluster.

Example 10.6. Associating a network resource with a cluster

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network id="da05ac09-00be-45a1-b0b5-4a6a2438665f">
    <name>ovirtmgmt</name>
</network>

HTTP/1.1 201 Created
Location: http://{host}/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks/da05ac09-00be-45a1-b0b5-4a6a2438665f
Content-Type: application/xml

<network id="da05ac09-00be-45a1-b0b5-4a6a2438665f"
  href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks/
  da05ac09-00be-45a1-b0b5-4a6a2438665f">
    <name>ovirtmgmt</name>
    <status>
        <state>operational</state>
    </status>
    <description>Display Network</description>
    <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"
```

```
      href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95"/>
    <data_center id="d70d5e2d-b8ad-494a-a4d2-c7a5631073c4"
      href="/ovirt-engine/api/datacenters/d70d5e2d-b8ad-494a-a4d2-
c7a5631073c4"/>
    <required>true</required>
    <usages>
        <usage>VM</usage>
    </usages>
</network>
```

Update the resource with a **PUT** request.

Example 10.7. Setting the display network status

```
PUT /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks/da05ac09-00be-45a1-b0b5-4a6a2438665f HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network>
    <required>false</required>
    <usages>
        <usage>VM</usage>
        <usage>DISPLAY</usage>
    </usages>
</network>
```

The required or optional network status is set using a **PUT** request to specify the Boolean value (true or false) of the **required** element.

Example 10.8. Setting optional network status

```
PUT /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks/da05ac09-00be-45a1-b0b5-4a6a2438665f HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network>
    <required>false</required>
</network>
```

An association is removed with a **DELETE** request to the appropriate element in the collection.

Example 10.9. Removing a network association from a cluster

```
DELETE /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks/da05ac09-00be-45a1-b0b5-4a6a2438665f HTTP/1.1

HTTP/1.1 204 No Content
```

10.7.2. Storage Volumes Sub-Collection

10.7.2.1. Red Hat Gluster Storage Volumes Sub-Collection

Red Hat Virtualization provides a means for creating and managing Red Hat Gluster Storage volumes. Red Hat Gluster Storage volumes are associated with clusters and are represented with the **`glustervolumes`** sub-collection.

The representation of a Red Hat Gluster Storage volume resource in the **`glustervolumes`** sub-collection is defined using the following elements:

Table 10.5. Gluster volume elements

Element	Type	Description	Properties
`volume_type`	enumerated	Defines the volume type. See the **`capabilities`** collection for a list of volume types.	
`bricks`	relationship	The sub-collection for the Red Hat Gluster Storage bricks. When creating a new volume, the request requires a set of **`brick`** elements to create and manage in this cluster. Requires the **`server_id`** of the Red Hat Gluster Storage server and a **`brick_dir`** element for the brick directory	
`transport_types`	complex	Defines a set of volume **`transport_type`** elements. See the **`capabilities`** collection for a list of available transport types.	
`replica_count`	integer	Defines the file replication count for a replicated volume.	
`stripe_count`	integer	Defines the stripe count for a striped volume	
`options`	complex	A set of additional Red Hat Gluster Storage **`option`** elements. Each **`option`** includes an option **`name`** and a **`value`**.	

Example 10.10. An XML representation of a Red Hat Gluster Storage volume

```
<gluster_volume id="99408929-82cf-4dc7-a532-9d998063fa95"
```

```
  href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95
  /glustervolume/e199f877-900a-4e30-8114-8e3177f47651">
    <name>GlusterVolume1</name>
    <link rel="bricks"
      href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95
      /glustervolume/e199f877-900a-4e30-8114-8e3177f47651/bricks"/>
    <volume_type>DISTRIBUTED_REPLICATE</volume_type>
    <transport_types>
        <transport_type>TCP</transport_type>
    </transport_types>
    <replica_count>2</replica_count>
    <stripe_count>1</stripe_count>
    <options>
        <option>
            <name>cluster.min-free-disk</name>
            <value>536870912</value>
        </option>
    </options>
</gluster_volume>
```

Create a Red Hat Gluster Storage volume via a **POST** request with the required **name**, **volume_type** and **bricks** to the sub-collection.

Example 10.11. Creating a Red Hat Gluster Storage volume

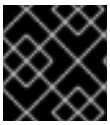
```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<gluster_volume>
    <name>GlusterVolume1</name>
    <volume_type>DISTRIBUTED_REPLICATE</volume_type>
    <bricks>
        <brick>
            <server_id>server1</server_id>
            <brick_dir>/exp1</brick_dir>
        </brick>
    <bricks>
</gluster_volume>
```

Remove a Red Hat Gluster Storage volume with a **DELETE** request.

Example 10.12. Removing a Red Hat Gluster Storage volume

```
DELETE /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-8e3177f47651
HTTP/1.1

HTTP/1.1 204 No Content
```

–



IMPORTANT

Resources in the **glustervolumes** sub-collection cannot be updated.

10.7.2.2. Bricks Sub-Collection

The **glustervolumes** sub-collection contains its own **bricks** sub-collection to define individual bricks in a Red Hat Gluster Storage volume. Additional information can be retrieved for **GET** requests using the **All-Content: true** header.

The representation of a volume's **bricks** sub-collection is defined using the following elements:

Table 10.6. Brick elements

Element	Type	Description	Properties
server_id	string	A reference to the Red Hat Gluster Storage server.	
brick_dir	string	Defines a brick directory on the Red Hat Gluster Storage server.	
replica_count	integer	Defines the file replication count for the brick in the volume.	
stripe_count	integer	Defines the stripe count for the brick in the volume	

Create new bricks via a **POST** request with the required **server_id** and **brick_dir** to the sub-collection.

Example 10.13. Adding a brick

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-8e3177f47651/bricks
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<brick>
    <server_id>server1</server_id>
    <brick_dir>/exp1</brick_dir>
</brick>
```

Remove a brick with a **DELETE** request.

Example 10.14. Removing a brick

```
DELETE /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-
8e3177f47651/bricks/0a473ebe-01d2-444d-8f58-f565a436b8eb HTTP/1.1

HTTP/1.1 204 No Content
```

IMPORTANT

Resources in the **bricks** sub-collection cannot be updated.

10.7.2.3. Actions

10.7.2.3.1. Start Action

The **start** action makes a Gluster volume available for use.

Example 10.15. Starting a Volume

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-8e3177f47651/start
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<action/>
```

Use an optional **force** Boolean element to force the action for a running volume. This is useful for starting disabled brick processes in a running volume.

10.7.2.3.2. Stop Action

The **stop** action deactivates a Gluster volume.

Example 10.16. Stopping a Volume

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-8e3177f47651/stop
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<action/>
```

Use an optional **force** Boolean element to brute force the stop action.

10.7.2.3.3. Set Option Action

The **`setoption`** action sets a volume option.

Example 10.17. Set an option

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-
8e3177f47651/setoption HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<action>
    <option>
        <name>cluster.min-free-disk</name>
        <value>536870912</value>
    </option>
</action>
```

10.7.2.3.4. Reset Option Action

The **`resetoption`** action resets a volume option.

Example 10.18. Reset an option

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-
8e3177f47651/resetoption HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<action>
        <option>
            <name>cluster.min-free-disk</name>
        </option>
</action>
```

10.7.2.3.5. Reset All Options Action

The **`resetalloptions`** action resets all volume options.

Example 10.19. Reset all options

```
POST /ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/e199f877-900a-4e30-8114-
8e3177f47651/resetalloptions HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<action/>
```

10.7.3. Affinity Groups Sub-Collection

10.7.3.1. Affinity Group Sub-Collection

The representation of a virtual machine affinity group resource in the **affinitygroups** sub-collection is defined using the following elements:

Table 10.7. Affinity group elements

Element	Type	Description	Properties
name	string	A plain text, human readable name for the affinity group.	
cluster	relationship	A reference to the cluster to which the affinity group applies.	
positive	Boolean: true or false	Specifies whether the affinity group applies positive affinity or negative affinity to virtual machines that are members of that affinity group.	
enforcing	Boolean: true or false	Specifies whether the affinity group uses hard or soft enforcement of the affinity applied to virtual machines that are members of that affinity group.	

Example 10.20. An XML representation of a virtual machine affinity group

```
<affinity_group href="/ovirt-engine/api/clusters/00000000-0000-0000-
0000-000000000000/affinitygroups/00000000-0000-0000-0000-000000000000"
id="00000000-0000-0000-0000-000000000000">
  <name>AF_GROUP_001</name>
  <cluster href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000"
    id="00000000-0000-0000-0000-000000000000"/>
  <positive>true</positive>
  <enforcing>true</enforcing>
</affinity_group>
```

Create a virtual machine affinity group via a **POST** request with the required **name** attribute.

Example 10.21. Creating a virtual machine affinity group

```
POST https://XX.XX.XX.XX/ovirt-engine/api/clusters/00000000-0000-0000-
0000-000000000000/affinitygroups HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<affinity_group>
```

```
  <name>AF_GROUP_001</name>
  <positive>true</positive>
  <enforcing>true</enforcing>
</affinity_group>
```

Remove a virtual machine affinity group with a **DELETE** request.

Example 10.22. Removing a virtual machine affinity group

```
DELETE https://XX.XX.XX.XX/ovirt-engine/api/clusters/00000000-0000-0000-
0000-000000000000/affinitygroups/00000000-0000-0000-0000-000000000000
HTTP/1.1

HTTP/1.1 204 No Content
```

CHAPTER 11. NETWORKS

11.1. NETWORK ELEMENTS

The **networks** collection provides information about the logical networks in a Red Hat Virtualization environment. An API user accesses this information through the **rel="networks"** link obtained from the entry point URI.

The following table shows specific elements contained in a network resource representation.

Table 11.1. Network elements

Element	Type	Description	Properties
link rel="vnicprofiles"	relationship	A link to the sub-collection for VNIC profiles attached to this logical network.	
link rel="labels"	relationship	A link to the sub-collection for labels attached to this logical network.	
data_center id=	GUID	A reference to the data center of which this cluster is a member.	
vlan id=	integer	A VLAN tag.	
stp	Boolean: true or false	**true** if Spanning Tree Protocol is enabled on this network.	
mtu	integer	Sets the maximum transmission unit for the logical network. If omitted, the logical network uses the default value.	
status	One of **operational** or **non_operational**	The status of the network. These states are listed in **network_states** under **capabilities**.	
usages	complex	Defines a set of **usage** elements for the network. Users can define networks as **VM** networks at this level.	

IMPORTANT

The API as documented in this section is experimental and subject to change. It is not covered by the backwards compatibility statement.

11.2. XML REPRESENTATION OF A NETWORK RESOURCE

Example 11.1. An XML representation of a network resource

```
<network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000"
  id="00000000-0000-0000-0000-000000000000">
  <name>ovirtmgmt</name>
  <description>Management Network</description>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/permissions" rel="permissions"/>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/vnicprofiles" rel="vnicprofiles"/>
  <link href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/labels" rel="labels"/>
  <data_center href="/ovirt-engine/api/datacenters/00000000-0000-0000-
0000-000000000000"
    id="00000000-0000-0000-0000-000000000000"/>
  <stp>false</stp>
  <mtu>0</mtu>
  <usages>
    <usage>vm</usage>
  </usages>
</network>
```

11.3. JSON REPRESENTATION OF A NETWORK RESOURCE

Example 11.2. A JSON representation of a network resource

```
{
  "network" : [ {
    "data_center" : {
      "href" : "/ovirt-engine/api/datacenters/00000002-0002-0002-0002-
000000000255",
      "id" : "00000002-0002-0002-0002-000000000255"
    },
    "stp" : "false",
    "mtu" : "0",
    "usages" : {
      "usage" : [ "vm" ]
    },
    "name" : "ovirtmgmt",
    "description" : "Management Network",
    "href" : "/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009",
    "id" : "00000000-0000-0000-0000-000000000009",
    "link" : [ {
      "href" : "/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009/permissions",
      "rel" : "permissions"
    }, {
      "href" : "/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009/vnicprofiles",
      "rel" : "vnicprofiles"
    }, {
```

```
      "href" : "/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009/labels",
      "rel" : "labels"
    } ]
  } ]
}
```

11.4. METHODS

11.4.1. Creating a Network Resource

Creation of a new network requires the **name** and **datacenter** elements.

Example 11.3. Creating a network resource

```
POST /ovirt-engine/api/networks HTTP/1.1
Accept: application/xml
Content-type: application/xml

<network>
    <name>network 1</name>
    <data_center id="00000000-0000-0000-0000-000000000000"/>
</network>
```

11.4.2. Updating a Network Resource

The **name**, **description**, **ip**, **vlan**, **stp** and **display** elements are updatable post-creation.

Example 11.4. Updating a network resource

```
PUT /ovirt-engine/api/networks/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<network>
    <description>Network 1</description>
</network>
```

11.4.3. Removing a Network Resource

Removal of a network requires a **DELETE** request.

Example 11.5. Removing a network

```
DELETE /ovirt-engine/api/networks/00000000-0000-0000-0000-000000000000
HTTP/1.1
```

```
HTTP/1.1 204 No Content
```

11.5. SUB-COLLECTIONS

11.5.1. Network VNIC Profile Sub-Collection

VNIC (Virtual Network Interface Controller) profiles, also referred to as virtual machine interface profiles, are customized profiles applied to users and groups to limit network bandwidth. Each **vnicprofile** contains the following elements:

Table 11.2. Elements for vnic profiles

Element	Type	Description
name	string	The unique identifier for the profile.
description	string	A plain text description of the profile.
network	string	The unique identifier of the logical network to which the profile applies.
port_mirroring	Boolean: **true** or **false**	The default is **false**.

Example 11.6. An XML representation of the network's vnicprofile sub-collection

```
<vnic_profile href= "/ovirt-engine/api/vnicprofiles/f9c2f9f1-3ae2-4100-
a9a5-285ebb755c0d" id="f9c2f9f1-3ae2-4100-a9a5-285ebb755c0d">
 <name>Peanuts</name>
 <description>shelled</description>
 <network href= "/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009" id="00000000-0000-0000-0000-000000000009"/>
 <port_mirroring>false</port_mirroring>
 </vnic_profile>
</vnic_profiles>
```

11.5.2. Network Labels Sub-Collection

Network labels are plain text, human-readable labels that allow you to automate the association of logical networks with physical host network interfaces. Each **label** contains the following elements:

Table 11.3. Elements for labels

Element	Type	Description
network	string	The **href** and **id** of the networks to which the label is attached.

Example 11.7. An XML representation of the network's labels sub-collection

```
<labels>
  <label href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/labels/eth0" id="eth0">
    <network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000"
      id="00000000-0000-0000-0000-000000000000"/>
  </label>
</labels>
```

11.5.3. Methods

11.5.3.1. Attach Label to Logical Network Action

You can attach labels to a logical network to automate the association of that logical network with physical host network interfaces to which the same label has been attached.

Example 11.8. Action to attach a label to a logical network

```
POST /ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/labels/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<label id="Label_001" />
```

11.5.3.2. Removing a Label From a Logical Network

Removal of a label from a logical network requires a **DELETE** request.

Example 11.9. Removing a label from a logical network

```
DELETE /ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000/labels/[label_id] HTTP/1.1

HTTP/1.1 204 No Content
```

CHAPTER 12. STORAGE DOMAINS

12.1. STORAGE DOMAIN ELEMENTS

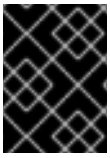
The **storagedomains** collection provides information about the storage domains in a Red Hat Virtualization environment. An API user accesses this information through the **rel="storagedomains"** link obtained from the entry point URI.

The following table shows specific elements contained in a storage domain resource representation.

Table 12.1. Storage domain elements

Element	Type	Description	Properties
link rel="permissions"	relationship	A link to the sub-collection for storage domain permissions.	
link rel="files"	relationship	A link to the **files** sub-collection for this storage domains.	
link rel="vms"	relationship	A link to the **vms** sub-collection for a storage domain with **type** set to **export**.	
link rel="templates"	relationship	A link to the **templates** sub-collection for a storage domain with **type** set to **export**.	
type	enumerated	The storage domain type. A list of enumerated values are available in **capabilities**.	
external_status	complex/enumerated	The storage domain health status as reported by external systems and plug-ins. The **state** element contains an enumerated value of **ok**, **info**, **warning**, **error**, or **failure**.	
master	Boolean: true or false	**true** if this is the master storage domain of a data center.	
host	complex	A reference to the host on which this storage domain should be initialized. The only restriction on this host is that it should have access to the physical storage specified.	

Element	Type	Description	Properties
storage	complex	Describes the underlying storage of the storage domain.	
available	integer	Space available in bytes.	
used	integer	Space used in bytes.	
committed	integer	Space committed in bytes.	
storage_format	enumerated	Describes the storage format version for the storage domain. A list of enumerated values are available in **capabilities**.	
wipe_after_delete	Boolean: true or false	Sets the wipe after delete option by default on the storage domain. This option can be edited after the domain is created, but doing so will not change the wipe after delete property of disks that already exist.	
warning_low_space_indicator	integer	A percentage value that sets the warning low space indicator option. If the free space available on the storage domain is below this percentage, warning messages are displayed to the user and logged.	
critical_space_action_blocker	integer	A value in GB that sets the critical space action blocker option. If the free space available on the storage domain is below this value, error messages are displayed to the user and logged, and any new action that consumes space, even temporarily, will be blocked.	



IMPORTANT

The API as documented in this chapter is experimental and subject to change. It is not covered by the backwards compatibility statement.

12.2. XML REPRESENTATION OF A STORAGE DOMAIN

Example 12.1. An XML representation of a storage domain

```
<storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"
  href="/ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed">
    <name>data0</name>
    <link rel="permissions"
      href="/ovirt-engine/api/storagedomains/be24cd98-8e23-49c7-b425-
1a12bd12abb0/permissions"/>
    <link rel="files"
      href="/ovirt-engine/api/storagedomains/be24cd98-8e23-49c7-b425-
1a12bd12abb0/files"/>
    <type>data</type>
    <master>true</master>
    <storage>
        <type>nfs</type>
        <address>172.31.0.6</address>
        <path>/exports/RHEVX/images/0</path>
    </storage>
    <available>156766306304</available>
    <used>433791696896</used>
    <committed>617401548800</committed>
    <storage_format>v1</storage_format>
    <wipe_after_delete>true</wipe_after_delete>
    <warning_low_space_indicator>10</warning_low_space_indicator>
    <critical_space_action_blocker>5</critical_space_action_blocker>
</storage_domain>
```

12.3. JSON REPRESENTATION OF A STORAGE DOMAIN

Example 12.2. A JSON representation of a storage domain

```
{
  "storage_domain" : [ {
    "type" : "data",
    "master" : "false",
    "storage" : {
      "address" : "192.0.2.0",
      "type" : "nfs",
      "path" : "/storage/user/nfs"
    },
    "available" : 193273528320,
    "used" : 17179869184,
    "committed" : 0,
    "storage_format" : "v3",
    "name" : "NFS_01",
    "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-a7ee-
c6fd4718aaba",
    "id" : "8827b158-6d2e-442d-a7ee-c6fd4718aaba",
    "link" : [ {
      "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-
a7ee-c6fd4718aaba/permissions",
      "rel" : "permissions"
```

```
    }, {
      "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-
a7ee-c6fd4718aaba/disks",
      "rel" : "disks"
    }, {
      "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-
a7ee-c6fd4718aaba/storageconnections",
      "rel" : "storageconnections"
    }, {
      "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-
a7ee-c6fd4718aaba/disksnapshots",
      "rel" : "disksnapshots"
    }, {
      "href" : "/ovirt-engine/api/storagedomains/8827b158-6d2e-442d-
a7ee-c6fd4718aaba/diskprofiles",
      "rel" : "diskprofiles"
    } ]
  } ]
}
```

12.4. METHODS

12.4.1. Creating a Storage Domain

Creation of a new storage domain requires the **name**, **type**, **host** and **storage** elements. Identify the **host** element with the **id** attribute or **name** element.

You can enable the wipe after delete option by default on the storage domain by specifying **<wipe_after_delete>** in the **POST** request. This option can be edited after the domain is created, but doing so will not change the wipe after delete property of disks that already exist.

Example 12.3. Creating a storage domain

```
POST /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
    <name>data1</name>
    <type>data</type>
    <host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"/>
    <storage>
        <type>nfs</type>
        <address>172.31.0.6</address>
        <path>/exports/RHEVX/images/0</path>
    </storage>
</storage_domain>
```

The API user attaches the storage domain to a data center after creation.

12.4.2. Updating a Storage Domain

Only the **name** and **wipe after delete** elements are updatable post-creation. Changing the **wipe after delete** element will not change the wipe after delete property of disks that already exist.

Example 12.4. Updating a storage domain

```
PUT /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
    <name>data2</name>
    ...
    <wipe_after_delete>true</wipe_after_delete>
    ...
</storage_domain>
```

12.4.3. Removing a Storage Domain

Removal of a storage domain requires a **DELETE** request.

Example 12.5. Removing a storage domain

```
DELETE /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed HTTP/1.1

HTTP/1.1 204 No Content
```

12.5. STORAGE TYPES

12.5.1. Storage Types

The **storage** element contains a **type** element, which is an enumerated value found under the **capabilities** collection.

The storage element also contains additional elements specific to each storage **type**. The next few sections examine these additional storage **type** elements.

12.5.2. NFS Storage

The following table contains **nfs** specific elements in a **storage** description.

Table 12.2. NFS specific elements

Element	Type	Description	Properties
address	string	The host name or IP address of the NFS server.	

Element	Type	Description	Properties
path	string	The path of NFS mountable directory on the server.	

12.5.3. PosixFS Storage

The following table contains **posixfs** specific elements in a **storage** description.

Table 12.3. PosixFS specific elements

Element	Type	Description	Properties
address	string	The host name or IP address of the PosixFS server.	
path	string	The path of PosixFS mountable directory on the server.	
vfs_type	string	The Linux-supported file system type of the PosixFS share.	
mount_options	string	The options for mounting the PosixFS share.	

12.5.4. iSCSI and FCP Storage

The following table contains **iscsi** and **fcp** specific elements in a **storage** description.

Table 12.4. iSCSI and FCP specific elements

Element	Type	Description	Properties
logical_unit id=	complex	The **id** of the logical unit. A storage domain also accepts multiple iSCSI or FCP logical units.	
override_luns	Boolean	Defines whether to replace all logical unit settings with new settings. Set to **true** to override.	

The **logical_unit** contains a set of sub-elements.

Table 12.5. Logical unit elements

Element	Type	Description	Properties
address	string	The address of the server containing the storage device.	
port	integer	The port number of the server.	
target	string	The target IQN for the storage device.	
username	string	A CHAP user name for logging into a target.	
password	string	A CHAP password for logging into a target.	
serial	string	The serial ID for the target.	
vendor_id	string	The vendor name for the target.	
product_id	string	The product code for the target.	
lun_mapping	integer	The Logical Unit Number device mapping for the target.	

In the case of iSCSI, if a **logical_unit** description also contains details of the iSCSI target with the LUN in question, the target performs an automatic login when the storage domain is created.

12.5.5. LocalFS Storage

The **localfs** specific elements in a **storage** description are:

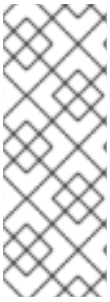
Table 12.6. Localfs specific elements

Element	Type	Description	Properties
path	string	The path of local storage domain on the host.	

A **localfs** storage domain requires a data center with **storage_type** set to **localfs**. This data center only contains a single host cluster, and the host cluster only contains a single host.

12.6. EXPORT STORAGE DOMAINS

12.6.1. Export Storage Domains



NOTE

The export storage domain is deprecated. Storage data domains can be unattached from a data center and imported to another data center in the same environment, or in a different environment. Virtual machines, floating virtual disk images, and templates can then be uploaded from the imported storage domain to the attached data center. See the Importing Existing Storage Domains section in the *Red Hat Virtualization Administration Guide* for information on importing storage domains.

Storage domains with **type** set to **export** contain **vms** and **templates** sub-collections, which list the import candidate VMs and templates stored on that particular storage domain.

Example 12.6. Listing the virtual machines sub-collection of an export storage domain

```
GET /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/vms
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<vms>
    <vm id="082c794b-771f-452f-83c9-b2b5a19c0399"
      href="/ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/
      vms/082c794b-771f-452f-83c9-b2b5a19c0399">
        <name>vm1</name>
        ...
        <storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"
          href="/ovirt-engine/api/storagedomains/fabe0451-701f-4235-
8f7e-e20e458819ed"/>
        <actions>
            <link rel="import" href="/ovirt-engine/api/storagedomains/
              fabe0451-701f-4235-8f7e-e20e458819ed/vms/
              082c794b-771f-452f-83c9-b2b5a19c0399/import"/>
        </actions>
    </vm>
</vms>
```

VMs and templates in these collections have a similar representation to their counterparts in the top-level VMs and templates collection, except they also contain a **storage_domain** reference and an **import** action.

The **import** action imports a virtual machine or a template from an **export** storage domain. The destination cluster and storage domain is specified with **cluster** and **storage_domain** references.

Include an optional **name** element to give the virtual machine or template a specific name.

Example 12.7. Action to import a virtual machine from an export storage domain

```
POST /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/vms/
082c794b-771f-452f-83c9-b2b5a19c0399/import HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>images0</name>
    </storage_domain>
    <cluster>
        <name>Default</name>
    </cluster>
</action>
```

Example 12.8. Action to import a template from an export storage domain

```
POST /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/templates/
082c794b-771f-452f-83c9-b2b5a19c0399/import HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>images0</name>
    </storage_domain>
    <cluster>
        <name>Default</name>
    </cluster>
</action>
```

Include an optional **clone** Boolean element to import the virtual machine as a new entity.

Example 12.9. Action to import a virtual machine as a new entity

```
POST /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/vms/
082c794b-771f-452f-83c9-b2b5a19c0399/import HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>images0</name>
    </storage_domain>
    <cluster>
        <name>Default</name>
```

```
    </cluster>
    <clone>true</clone>
    <vm>
        <name>MyVM</name>
    </vm>
    ...
</action>
```

Include an optional **disks** element to choose which disks to import using individual **disk id** elements.

Example 12.10. Selecting disks for an import action

```
POST /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/vms/
082c794b-771f-452f-83c9-b2b5a19c0399/import HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <cluster>
        <name>Default</name>
    </cluster>
    <vm>
        <name>MyVM</name>
    </vm>
    ...
    <disks>
        <disk id="4825ffda-a997-4e96-ae27-5503f1851d1b"/>
    </disks>
</action>
```

Delete a virtual machine or template from an **export** storage domain with a **DELETE** request.

Example 12.11. Delete virtual machine from an export storage domain

```
DELETE /ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed/vms/
082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml

HTTP/1.1 204 No Content
```

12.7. GLANCE IMAGE STORAGE DOMAINS

12.7.1. Glance Image Storage Domains

Storage domains with type set to **Image** represent instances of an OpenStack image service that has been added to the Red Hat Virtualization environment as an external provider. These Glance image storage domains contain an **images** sub-collection with virtual machine images that have been exported

to or can be imported from that Glance image storage domain.

Example 12.12. Listing the images sub-collection of a Glance image storage domain

```
GET /ovirt-engine/api/storagedomains/00000000-0000-0000-0000-
000000000000/images
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<images>
  <image href="/ovirt-engine/api/storagedomains/00000000-0000-0000-0000-
000000000000/images/
    00000000-0000-0000-0000-000000000000" id="00000000-0000-0000-0000-
000000000000">
    <actions>
      <link href="/ovirt-engine/api/storagedomains/00000000-0000-0000-
0000-000000000000/images/
        00000000-0000-0000-0000-000000000000/import" rel="import"/>
    </actions>
    <name>RHEL_65_Disk_001</name>
    <storage_domain href="/ovirt-engine/api/storagedomains/00000000-
0000-0000-0000-000000000000"
      id="00000000-0000-0000-0000-000000000000"/>
  </image>
  <image href="/ovirt-engine/api/storagedomains/00000000-0000-0000-0000-
000000000000/images/
    00000000-0000-0000-0000-000000000000" id="00000000-0000-0000-0000-
000000000000">
    <actions>
      <link href="/ovirt-engine/api/storagedomains/00000000-0000-0000-
0000-000000000000/images/
        00000000-0000-0000-0000-000000000000/import" rel="import"/>
    </actions>
    <name>RHEL_65_Disk_002</name>
    <storage_domain href="/ovirt-engine/api/storagedomains/00000000-
0000-0000-0000-000000000000"
      id="00000000-0000-0000-0000-000000000000"/>
  </image>
</images>
```

The **`import`** action imports a virtual machine image from a Glance image storage domain. The destination storage domain is specified with a **`storage_domain`** reference, and the destination cluster with a **`cluster`** reference.

Include an optional **`name`** element to give the virtual machine or template a specific name.

Example 12.13. Action to import a virtual machine from a Glance image storage domain

```
POST /ovirt-engine/api/storagedomains/00000000-0000-0000-
000000000000/images/
00000000-0000-0000-000000000000/import HTTP/1.1
```

```
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>images0</name>
    </storage_domain>
    <cluster>
        <name>images0</name>
    </cluster>
</action>
```

You can also import images as templates by specifying the **import_as_template** reference:

Example 12.14. Action to import a virtual machine from a Glance image storage domain as a template

```
POST /ovirt-engine/api/storagedomains/00000000-0000-0000-
000000000000/images/
00000000-0000-0000-000000000000/import HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>images0</name>
    </storage_domain>
    <cluster>
        <name>images0</name>
    </cluster>
    </import_as_template>true</import_as_template>
</action>
```

12.8. IMPORTING A BLOCK STORAGE DOMAIN

12.8.1. Importing a Block Storage Domain

An existing block storage domain with **type** set to **iscsi** or **fcp** can be imported to the engine using the REST API. The ability to import storage domains allows you to recover data in the event of a failure in the engine database, and to migrate data from one data center or environment to another.

This procedure assumes the storage domain is not attached to a data center or host in any environment. To import and attach an existing block storage domain to a data center, the target data center must be initialized.

Procedure 12.1. Importing a block storage domain

1. Discover the targets on your iSCSI storage server:

```
POST /ovirt-engine/api/hosts/052a880a-53e0-4fe3-9ed5-
01f939d1df66/iscsidiscover
```

```
Accept: application/xml
Content-Type: application/xml

<action>
    <iscsi>
        <address>192.0.2.0</address>
        <port>3260</port>
    </iscsi>
</action>
```

2. Get a list of storage domains that are candidates to be imported, using the iSCSI targets discovered in the previous step:

```
POST /ovirt-engine/api/hosts/052a880a-53e0-4fe3-9ed5-
01f939d1df66/unregisteredstoragedomainsdiscover HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <iscsi>
        <address>192.0.2.0</address>
    </iscsi>
    <iscsi_target>iqn.name1.120.01</iscsi_target>
    <iscsi_target>iqn.name2.120.02</iscsi_target>
    <iscsi_target>iqn.name3.120.03</iscsi_target>
</action>
```

The response shows a list of storage domains not associated with a host, similar to the following:

```
<action>
    <iscsi>
        <address>192.0.2.0</address>
    </iscsi>
    <storage_domains>
        <storage_domain id="6ab65b16-0f03-4b93-85a7-5bc3b8d52be0">
            <name>scsi4</name>
            <type>data</type>
            <external_status>
              <state>ok</state>
            </external_status>
            <master>false</master>
            <storage>
                <type>iscsi</type>
                <volume_group id="OLkKwa-VmEM-abW7-hPiv-BGrw-sQ2E-
vTdAy1"/>
            </storage>
            <available>0</available>
            <used>0</used>
            <committed>0</committed>
            <storage_format>v3</storage_format>
        </storage_domain>
    <status>
        <state>complete</state>
    </status>
```

```
    <iscsi_target>iqn.name1.120.01</iscsi_target>
    <iscsi_target>iqn.name2.120.02</iscsi_target>
    <iscsi_target>iqn.name3.120.03</iscsi_target>
</action>
```

3. Import the iSCSI storage domains to the host:

```
POST /ovirt-engine/api/storagedomains/ HTTP/1.1
  Accept: application/xml
  Content-type: application/xml

<storage_domain id="6ab65b16-0f03-4b93-85a7-5bc3b8d52be0">
    <import>true</import>
    <host id="052a880a-53e0-4fe3-9ed5-01f939d1df66" />
    <type>data</type>
    <storage>
        <type>iscsi</type>
    </storage>
</storage_domain>
```

You have now imported the block storage domain to your host.

You may now wish to attach the storage domain to the host, and find any unregistered disks. Attach the storage domain and associated disks with the following steps:

Procedure 12.2. Attaching a block storage domain

1. Attach the storage domain to your data center:

```
POST /ovirt-engine/api/datacenters/01a45ff0-915a-45e0-8d56-
5253234ac988/storagedomains
Accept: application/xml
Content-Type: application/xml

<storage_domain>
    <name>scsi4</name>
</storage_domain>
```

2. Find the unregistered disks on the storage domain:

```
GET /ovirt-engine/api/storagedomains/6ab65b16-0f03-4b93-85a7-
5bc3b8d52be0/disks;unregistered
Accept: application/xml
Content-Type: application/xml
```

This will return information about any unregistered disks on the storage domain, with a response similar to:

```
<disk href= "/ovirt-engine/api/storagedomains/6ab65b16-0f03-4b93-
85a7-5bc3b8d52be0/disks/b662f6da-3e97-4bb6-8a50-bda9980a6e83"
id="b662f6da-3e97-4bb6-8a50-bda9980a6e83">
  <actions>
      <link href= "/ovirt-engine/api/storagedomains/6ab65b16-0f03-
```

```
4b93-85a7-5bc3b8d52be0/disks/b662f6da-3e97-4bb6-8a50-
bda9980a6e83/export" rel="export"/>
  </actions>
  <name>disk1</name>
  <description/>
  <link href= "/ovirt-engine/api/storagedomains/6ab65b16-0f03-4b93-
85a7-5bc3b8d52be0/disks/b662f6da-3e97-4bb6-8a50-
bda9980a6e83/permissions" rel="permissions"/>
  <link href= "/ovirt-engine/api/storagedomains/6ab65b16-0f03-4b93-
85a7-5bc3b8d52be0/disks/b662f6da-3e97-4bb6-8a50-
bda9980a6e83/statistics" rel="statistics"/>
  <alias>disk1</alias>
  <image_id>930d653e-2a11-45ce-8042-9935584a3f87</image_id>
  <storage_domain href= "/ovirt-engine/api/storagedomains/6ab65b16-
0f03-4b93-85a7-5bc3b8d52be0" id="8ac10ec5-7cc9-4b1c-9c97-
f121a9e4679a"/>
  <storage_domains>
      <storage_domain id="6ab65b16-0f03-4b93-85a7-5bc3b8d52be0"/>
  </storage_domains>
  <size>10737418240</size>
  <provisioned_size>10737418240</provisioned_size>
  <actual_size>10737418240</actual_size>
  <status>
      <state>ok</state>
  </status>
  <interface>ide</interface>
  <format>raw</format>
  <sparse>false</sparse>
  <bootable>false</bootable>
  <shareable>false</shareable>
  <wipe_after_delete>false</wipe_after_delete>
  <propagate_errors>false</propagate_errors>
  <storage_type>image</storage_type>
</disk>
```

3. Attach the disk to the storage domain:

```
POST /ovirt-engine/api/storagedomains/6ab65b16-0f03-4b93-85a7-
5bc3b8d52be0/disks;unregistered
Accept: application/xml
Content-Type: application/xml

<disk id='b662f6da-3e97-4bb6-8a50-bda9980a6e83'></disk>
```

The disk is now attached to the imported block storage domain.

12.9. SUB-COLLECTIONS

12.9.1. Files Sub-Collection

The **files** sub-collection under each storage domain provides a way for clients to list available files. This sub-collection is specifically targeted to ISO storage domains, which contain ISO images and virtual floppy disks (VFDs) that an administrator uploads through Red Hat Virtualization Manager.

The addition of a CD-ROM device to a VM requires an ISO image from the **files** sub-collection of an ISO storage domain.

Example 12.15. Listing the files sub-collection of an ISO storage domain

```
GET /ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/files HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<files>
    <file id="en_winxp_pro_with_sp2.iso"
      href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/files/
      en_winxp_pro_with_sp2.iso">
        <name>en_winxp_pro_with_sp2.iso</name>
        <type>iso</type>
        <storage_domain id="00f0d9ce-da15-4b9e-9e3e-3c898fa8b6da"
          href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-
9e3e-3c898fa8b6da"/>
    </file>
    <file id="boot.vfd"
      href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-9e3e-
3c898fa8b6da/files/
      boot.vfd">
        <name>boot.vfd</name>
        <type>vfd</type>
        <storage_doman id="00f0d9ce-da15-4b9e-9e3e-3c898fa8b6da"
          href="/ovirt-engine/api/storagedomains/00f0d9ce-da15-4b9e-
9e3e-3c898fa8b6da"/>
    </file>
</files>
```

Like other resources, files have opaque **id** and **href** attributes. The **name** element contains the filename.

12.10. ACTIONS

12.10.1. Importing an Existing Storage Domain

NOTE

The export storage domain is deprecated. Storage data domains can be unattached from a data center and imported to another data center in the same environment, or in a different environment. Virtual machines, floating virtual disk images, and templates can then be uploaded from the imported storage domain to the attached data center. See the Importing Existing Storage Domains section in the *Red Hat Virtualization Administration Guide* for information on importing storage domains.

The API provides a user with the ability to remove an ISO or Export storage domain from one Red Hat

Virtualization Manager instance without re-formatting the underlying storage and import it into another instance. Importing is achieved similarly to adding a new storage domain, except the **name** is not specified.

Example 12.16. Importing an existing export storage domain

```
POST /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<storage_domain>
    <type>export</type>
    <storage>
        <type>nfs</type>
        <address>172.31.0.6</address>
        <path>/exports/RHEVX/export-domain</path>
    </storage>
    <host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"/>
</storage_domain>

HTTP/1.1 201 Created
Content-Type: application/xml

<storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"
  href="/ovirt-engine/api/storagedomains/fabe0451-701f-4235-8f7e-
e20e458819ed">
    <name>export1</name>
    ...
</storage_domain>
```

12.10.2. Deleting a Storage Domain

A **storage_domain** reference is passed in the body of a **DELETE** request for a storage domain. The **storage_domain** reference is in the following form:

```
<storage_domain>
    <host id="..."/>
</storage_domain>
```

OR

```
<storage_domain>
    <host>
        <name>...</name>
    </host>
</storage_domain>
```

Format Storage Domain

An API user provides a optional **format** element to specify whether or not to format the storage domain after deletion.

Example 12.17. Formatting a storage domain after deletion

```
<storage_domain>
    <host id="..."/>
    <format>true</format>
</storage_domain>
```

If no **format** element is passed, the storage domain remains unformatted.

Logical Removal of Storage Domain

The API also provides a function for the logical removal of the storage domain. This retains the storage domain's data for import. Use the **destroy** element to logically remove the storage domain and retain the data.

Example 12.18. Logical removal of a storage domain

```
<storage_domain>
    <host id="..."/>
    <destroy>true</destroy>
</storage_domain>
```

12.10.3. Refreshing the LUN Size

Users can refresh the LUN size after increasing the size of the underlying LUN on the storage server. The **refreshluns** action forces a rescan of the provided LUNs and updates the database with the new size if required.

Example 12.19. Refreshing the LUN Size

```
POST /ovirt-engine/api/storagedomains/262b056b-aede-40f1-9666-
b883eff59d40/refreshluns HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
   <logical_units>
      <logical_unit id="1IET_00010001"/>
      <logical_unit id="1IET_00010002"/>
   </logical_units>
</action>
```

CHAPTER 13. STORAGE CONNECTIONS

13.1. STORAGE CONNECTION ELEMENTS

Table 13.1. Storage Connection Base Elements

Element	Type	Description	Properties
`type`	One of `nfs`, `posixfs`, `local`, or `iscsi`	The type of storage domain.	
`address`	string	The hostname or IP address of the storage domain.	 (Only required for NFS and iSCSI)
`host`	string	The `id` or `name` of the hypervisor. The `host` is optional. Providing it will attempt a connection to the storage via the host; not providing it will lead to persisting storage details in the database.	

Table 13.2. Storage Connection File-based Storage Elements

Element	Type	Description	Properties
`path`	string	The mounted file path of the storage domain. The `path` cannot be updated to one already used by a storage connection.	
`mount_options`	string	The options for mounting the PosixFS share.	
`vfs_type`	string	The Linux-supported file system type of the PosixFS share.	 
`nfs_version`	string	The version of NFS used.	
`nfs_timeo`	integer	The amount of time, in deciseconds, the NFS client will wait for a request to complete.	

Element	Type	Description	Properties
`nfs_retrans`	integer	The number of retransmissions the NFS client will attempt to complete a request.	

Table 13.3. Storage Connection iSCSI elements

Element	Type	Description	Properties
`port`	integer	The TCP port used for the iSCSI storage domain.	
`target`	string	The target IQN for the storage device.	
`username`	string	A CHAP user name for logging into a target.	
`password`	string	A CHAP password for logging into a target.	

13.2. XML REPRESENTATION OF A STORAGE CONNECTION RESOURCE

Example 13.1. An XML representation of a storage connection resource

```
<storage_connections>
  <storage_connection href= "/ovirt-
engine/api/storageconnections/608c5b96-9939-4331-96b5-197f28aa2e35"
id="608c5b96-9939-4331-96b5-197f28aa2e35">
    <address>domain.example.com</address>
    <type>nfs</type>
    <path>/var/lib/exports/iso</path>
  </storage_connection>
  <storage_connection href= "/ovirt-
engine/api/storageconnections/2ebb3f78-8c22-4666-8df4-e4bb7fec6b3a"
id="2ebb3f78-8c22-4666-8df4-e4bb7fec6b3a">
    <address>domain.example.com</address>
    <type>posixfs</type>
    <path>/export/storagedata/username/data</path>
    <vfs_type>nfs</vfs_type>
  </storage_connection>
</storage_connections>
```

13.3. METHODS

13.3.1. Creating a New Storage Connection

Creating a new storage connection requires a **POST** request.

It is possible to create a new storage connection without adding a storage domain. The host **id** or **name** is optional; providing it will attempt a connection to the storage via the host.

Example 13.2. Creating a New Storage Connection

```
POST /ovirt-engine/api/storageconnections HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_connection>
   <type>nfs</type>
   <address>domain.example.com</address>
   <path>/export/storagedata/username/data</path>
   <host>
     <name>Host_Name</name>
   </host>
</storage_connection>
```

13.3.2. Deleting a Storage Connection

Deleting a storage connection requires a **DELETE** request. A storage connection can only be deleted if neither storage domain nor LUN disks reference it.

The host **name** or **id** is optional; providing it unmounts the connection from that host.

Example 13.3. Deleting Storage Connection

```
DELETE /ovirt-engine/api/storageconnections/Storage_Connection_ID
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host>
  <name>Host_Name</name>
</host>
```

13.3.3. Updating a Storage Connection

Updating an existing storage connection requires a **PUT** request. The storage domain must be in either maintenance mode or unattached to successfully update the connection.

Providing the host **name** or **id** is optional; if provided, the host attempts a connection to the updated storage details.

Example 13.4. Updating a Storage Connection

```
PUT /ovirt-engine/api/storageconnections/Storage_Connection_ID HTTP/1.1
```

```
Accept: application/xml
Content-type: application/xml

<storage_connection>
  <address>updated.example.domain.com</address>
  <host>
      <name>Host_name</name>
   </host>
</storage_connection>
```

13.3.4. Updating an iSCSI Storage Connection

Updating an existing iSCSI storage connection requires a **PUT** request. An iSCSI storage domain must be in maintenance mode or unattached to successfully update the connection.

Example 13.5. Updating a Storage Connection

```
PUT /ovirt-engine/api/storageconnections/Storage_Connection_ID HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_connection>
  <port>3456</port>
</storage_connection>
```

13.3.5. Adding New Storage Domain with Existing Storage Connection

Adding a new storage domain with existing storage connection requires a **POST** request. This is only applicable with file-based storage domains: **NFS**, **POSIX**, and **local**.

Example 13.6. Adding a New Storage Domain with Existing Storage Connection

```
POST /ovirt-engine/api/storagedomains HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_domain>
  <name>New_Domain</name>
  <type>data</type>
 <storage id="Storage_Connection_ID"/>
  <host>
    <name>Host_Name</name>
  </host>
</storage_domain>
```

13.3.6. Attaching an Additional Storage Connection to iSCSI Storage

Attaching an additional storage connection to an iSCSI storage domain requires a **POST** request.

Example 13.7. Attaching an Additional Storage Connection to iSCSI Storage

```
POST /ovirt-engine/api/storagedomains/iSCSI_Domain_ID/storageconnections
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storage_connection id="Storage_Connection_ID">
</storage_connection>
```

13.3.7. Detaching a Storage Connection from iSCSI Storage

Detaching a storage connection from an iSCSI storage domain requires a **DELETE** request.

Example 13.8. Detaching a Storage Connection from iSCSI Storage

```
DELETE /ovirt-
engine/api/storagedomains/iSCSI_Domain_ID/storageconnections/Storage_Con
nection_ID HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

13.3.8. Defining Credentials to an iSCSI Target

When an iSCSI storage domain is added using the Administration Portal, only a single user name and password can be specified for that domain. However, some setups require that each host in the cluster use a separate user name and password. Specific credentials can be applied to each iSCSI target per host by using the **storageconnectionextensions** element.

Example 13.9. Defining credentials to an iSCSI target

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/storageconnectionextensions HTTP/1.1
Accept: application/xml
Content-type: application/xml

<storageconnectionextension>
    <target>iqn.2010.05.com.example:iscsi.targetX</target>
    <username>jimmy</username>
    <password>p@55w0Rd!</password>
</storageconnectionextension>
```

CHAPTER 14. HOSTS

14.1. HOST ELEMENTS

The **hosts** collection provides information about the hosts in a Red Hat Virtualization environment. An API user accesses this information through the **rel="hosts"** link obtained from the entry point URI.

Additional information can be retrieved for **GET** requests using the **All-Content: true** header.

The following table shows specific elements contained in a host resource representation.

Table 14.1. Host elements

Element	Type	Description	Properties
link rel="storage"	relationship	A link to the **storage** sub-collection for host storage.	
link rel="nics"	relationship	A link to the **nics** sub-collection for host network interfaces.	
link rel="numanodes"	relationship	A link to the **numanodes** sub-collection for host NUMA nodes.	
link rel="tags"	relationship	A link to the **tags** sub-collection for host tags.	
link rel="permissions"	relationship	A link to the **permissions** sub-collection for host permissions.	
link rel="statistics"	relationship	A link to the **statistics** sub-collection for host statistics.	
link rel="hooks"	relationship	A link to the **hooks** sub-collection for host hooks.	
link rel="fenceagents"	relationship	A link to the **fenceagents** sub-collection for host fence agents.	
link rel="katelloerrata"	relationship	A link to the **katelloerrata** sub-collection for host errata.	
link rel="devices"	relationship	A link to the **devices** sub-collection for host devices.	

Element	Type	Description	Properties
link rel="networkattachments"	relationship	A link to the **networkattachments** sub-collection for host network configuration.	
link rel="unmanagednetworks"	relationship	A link to the **unmanagednetworks** sub-collection for unmanaged networks on the host.	
link rel="storageconnectionextensions"	relationship	A link to the **storageconnectionextensions** sub-collection for host storage connection extensions.	
name	string	The unique identifier for the host.	
root_password	string	The root password of this host, by convention only included in the client-provided host representation on creation.	
comment	string	Any comments regarding the host.	
address	string	The IP address or hostname of the host.	
certificate	complex	A reference to the host certificate details, including **organization** and **subject**.	
status	See below	The host status.	
external_status	complex/enumerated	The host health status as reported by external systems and plug-ins. The **state** element contains an enumerated value of **ok**, **info**, **warning**, **error**, or **failure**.	
cluster id=	GUID	A reference to the cluster that includes this host.	
port	integer	The listen port of the VDSM daemon running on this host.	

Element	Type	Description	Properties
type	One of **rhel** or **ovirt_node**	The host type.	
storage_manager priority=	Boolean: true or false	Specifies whether the host is a storage manager.	
version major= minor= build= revision= full_version=	complex	The compatibility level of the host.	
hardware_information	complex	Information regarding the hardware of the host, including **manufacturer**, **version**, **serial_number**, **product_name**, **uuid**, and **family**.	
power_management type=	complex	Configuration options for host power management, including **enabled**, **options**, **kdump_detection**, **automatic_pm_enabled**, and **agents**. See Section 14.4, “Power Management Elements” for more information on the host power management options.	
ksm	Boolean: true or false	**true** if Kernel SamePage Merging (KSM) is enabled.	
transparent_hugepages	Boolean: true or false	**true** if Transparent Hugepages is enabled.	
iscsi	complex	The SCSI **initiator** for the host.	
ssh	complex	Details regarding the SSH connection with the host, including **port** and **fingerprint**.	

Element	Type	Description	Properties
cpu	complex	Statistics for the host CPU. Includes sub-elements for the CPU's **name**, **topology cores=**, **topology sockets=**, **topology threads=** and **speed**. The **topology cores=** aggregates the total cores while the **topology sockets=** aggregates the total physical CPUs. The total cores available to virtual machines equals the number of sockets multiplied by the cores per socket.	
memory	integer	The total amount of host memory in bytes.	
max_scheduling_memory	integer	The maximum amount of memory that can be used in scheduling in bytes.	
summary	complex	Summary statistics of the virtual machines on the host. Includes sub-elements for numbers of **active**, **migrating** and **total** VMs.	
os type=	complex	Details regarding the operating system installed on the host, including **version full_version=**.	
libvirt_version major= minor= build= revision= full_version=	complex	The libvirt compatibility level of the host.	

The **status** contains one of the following enumerative values: **down**, **error**, **initializing**, **installing**, **install_failed**, **maintenance**, **non_operational**, **non_responsive**, **pending_approval**, **preparing_for_maintenance**, **connecting**, **reboot**, **unassigned** and **up**. These states are listed in **host_states** under **capabilities**.

14.2. XML REPRESENTATION OF A HOST

Example 14.1. An XML representation of a host

```
<host href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000">
    <actions>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/upgrade" rel="upgrade"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/setupnetworks" rel="setupnetworks"/>
```

```
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/fence" rel="fence"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/refresh" rel="refresh"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/install" rel="install"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/activate" rel="activate"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/deactivate" rel="deactivate"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/approve" rel="approve"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/forceselectspm" rel="forceselectspm"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/enrollcertificate" rel="enrollcertificate"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/iscsilogin" rel="iscsilogin"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/unregisteredstoragedomainsdiscover"
rel="unregisteredstoragedomainsdiscover"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/iscsidiscover" rel="iscsidiscover"/>
        <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/commitnetconfig" rel="commitnetconfig"/>
    </actions>
    <name>host1</name>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/storage" rel="storage"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics" rel="nics"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/numanodes" rel="numanodes"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/tags" rel="tags"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/permissions" rel="permissions"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/statistics" rel="statistics"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/hooks" rel="hooks"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/fenceagents" rel="fenceagents"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/katelloerrata" rel="katelloerrata"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/devices" rel="devices"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/networkattachments" rel="networkattachments"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/unmanagednetworks" rel="unmanagednetworks"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/storageconnectionextensions"
rel="storageconnectionextensions"/>
    <address>host1.example.com</address>
    <certificate>
```

```
        <organization>exampleorg</organization>
        <subject>O=exampleorg,CN=XX.XX.XX.XX</subject>
    </certificate>
    <status>
        <state>up</state>
    </status>
    <external_status>
        <state>ok</state>
    </external_status>
    <cluster href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000"/>
    <port>54321</port>
    <type>rhel</type>
    <storage_manager priority="2">false</storage_manager>
    <spm>
        <priority>2</priority>
        <status>
            <state>none</state>
        </status>
    </spm>
    <version major="4" minor="17" build="20" revision="0"
full_version="vdsm-4.17.20-0.el7ev"/>
    <power_management>
        <enabled>false</enabled>
        <pm_proxies/>
        <automatic_pm_enabled>true</automatic_pm_enabled>
        <kdump_detection>true</kdump_detection>
    </power_management>
    <ksm>
        <enabled>true</enabled>
    </ksm>
    <transparent_hugepages>
        <enabled>true</enabled>
    </transparent_hugepages>
    <iscsi>
        <initiator>iqn.2001-04.com.example:diskarrays-sn-
a8675309</initiator>
    </iscsi>
    <ssh>
        <port>22</port>

<fingerprint>00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00</fingerprin
t>
    </ssh>
    <cpu>
        <topology cores="2" sockets="1"/>
        <name>Intel(R) Xeon(R) CPU E5430 @ 2.66GHz</name>
        <speed>2656</speed>
    </cpu>
    <memory>12430868480</memory>
    <max_scheduling_memory>12026118144</max_scheduling_memory>
    <summary>
        <active>2</active>
        <migrating>0</migrating>
        <total>3</total>
    </summary>
```

```
    <protocol>stomp</protocol>
    <os type="RHEL">
        <version full_version="7.2-9.el7_2.1"/>
    </os>
    <libvirt_version major="1" minor="2" build="17" revision="0"
full_version="libvirt-1.2.17-13.el7_2.2"/>
    <kdump_status>disabled</kdump_status>
    <selinux>
        <mode>enforcing</mode>
    </selinux>
    <auto_numa_status>disable</auto_numa_status>
    <numa_supported>false</numa_supported>
    <live_snapshot_support>true</live_snapshot_support>
    <update_available>false</update_available>
    <device_passthrough>
        <enabled>true</enabled>
    </device_passthrough>
</host>
```

14.3. JSON REPRESENTATION OF A HOST

Example 14.2. A JSON representation of a host

```
{
  "host" : [ {
    "address" : "198.51.100.0",
    "certificate" : {
      "organization" : "example.com",
      "subject" : "O=example.com,CN=192.0.2.0"
    },
    "status" : {
      "state" : "up"
    },
    "cluster" : {
      "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb",
      "id" : "00000001-0001-0001-0001-0000000002fb"
    },
    "port" : "54321",
    "type" : "rhel",
    "storage_manager" : {
      "value" : "true",
      "priority" : "5"
    },
    "spm" : {
      "priority" : "5"
    },
    "version" : {
      "major" : "4",
      "minor" : "16",
      "build" : "8",
      "revision" : "1",
      "full_version" : "vdsm-4.16.8.1-6.el6ev"
    },
```

```
"hardware_information" : {
  "manufacturer" : "System Manufacturer To Be Filled By O.E.M.",
  "version" : "System Version To Be Filled By O.E.M.",
  "serial_number" : "Serial Number To Be Filled By O.E.M.",
  "product_name" : "Product Name To Be Filled By O.E.M.",
  "uuid" : "9fa0a1a2-a3a4-a5a6-a7a8-a9aaabacadae",
  "family" : "Family To Be Filled By O.E.M.",
  "supported_rng_sources" : {
    "source" : [ "RANDOM" ]
  }
},
"power_management" : {
  "enabled" : "false",
  "options" : {
    "option" : [ {
      "name" : "secure",
      "value" : "false"
    } ]
  },
  "automatic_pm_enabled" : "true",
  "kdump_detection" : "true",
  "type" : "apc"
},
"ksm" : {
  "enabled" : "false"
},
"transparent_hugepages" : {
  "enabled" : "true"
},
"iscsi" : {
  "initiator" : "iqn.1994-05.com.example:795610ff2632"
},
"ssh" : {
  "port" : "22",
  "fingerprint" : "77:27:38:25:8f:60:8d:93:9c:2c:b0:cb:5e:19:f4:53"
},
"cpu" : {
  "topology" : {
    "sockets" : "1",
    "cores" : "4",
    "threads" : "1"
  },
  "name" : "Intel(R) Core(TM)2 Quad CPU    Q9550  @ 2.83GHz",
  "speed" : 2833
},
"memory" : 2989490176,
"max_scheduling_memory" : 2584739840,
"summary" : {
  "active" : "0",
  "migrating" : "0",
  "total" : "0"
},
"protocol" : "stomp",
"os" : {
  "version" : {
    "full_version" : "6Server - 6.6.0.2.el6"
```

```
      },
      "type" : "RHEL"
    },
    "libvirt_version" : {
      "major" : "0",
      "minor" : "10",
      "build" : "2",
      "revision" : "0",
      "full_version" : "libvirt-0.10.2-46.el6_6.2"
    },
    "kdump_status" : "disabled",
    "selinux" : {
      "mode" : "enforcing"
    },
    "auto_numa_status" : "unknown",
    "numa_supported" : "false",
    "live_snapshot_support" : "true",
    "actions" : {
      "link" : [ {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/fence",
        "rel" : "fence"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/approve",
        "rel" : "approve"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/forceselectspm",
        "rel" : "forceselectspm"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/iscsilogin",
        "rel" : "iscsilogin"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/iscsidiscover",
        "rel" : "iscsidiscover"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/commitnetconfig",
        "rel" : "commitnetconfig"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/deactivate",
        "rel" : "deactivate"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/install",
        "rel" : "install"
      }, {
        "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/activate",
        "rel" : "activate"
      } ]
    },
```

```
    "name" : "Host-07",
    "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe",
    "id" : "ea7aa772-d2af-4a5c-9350-d86f005c93fe",
    "link" : [ {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/storage",
      "rel" : "storage"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/nics",
      "rel" : "nics"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/numanodes",
      "rel" : "numanodes"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/tags",
      "rel" : "tags"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/permissions",
      "rel" : "permissions"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/statistics",
      "rel" : "statistics"
    }, {
      "href" : "/ovirt-engine/api/hosts/ea7aa772-d2af-4a5c-9350-
d86f005c93fe/hooks",
      "rel" : "hooks"
    } ]
  } ]
}
```

14.4. POWER MANAGEMENT ELEMENTS

The **power_management** element provides users with the ability to set a power management configuration, which is required for host fencing. Certain sub-elements are required when configuring **power_management**.

Table 14.2. Power management options

Element	Type	Description	Properties
type=	fencing device code	A list of valid fencing device codes are available in the **capabilities** collection.	 

Element	Type	Description	Properties
enabled	Boolean: true or false	Indicates whether power management configuration is enabled or disabled.	
address	string	The host name or IP address of the host.	
username	string	A valid user name for power management.	
password	string	A valid, robust password for power management.	
options	complex	Fencing options for the selected **type=** specified with the option **name=""** and **value=""** strings.	
agents	complex	Specifies fence agent options when multiple fences are used. Use the **order** sub-element to prioritize the fence agents. Agents are run sequentially according to their order until the fence action succeeds. When two or more fence agents have the same **order**, they are run concurrently. Other sub-elements include **type**, **ip**, **user**, **password**, and **options**.	
automatic_pm_enabled	Boolean: true or false	Toggles the automated power control of the host in order to save energy. When set to **true**, the host will be automatically powered down if the cluster's load is low, and powered on again when required. This is set to **true** when a host is created, unless disabled by the user.	
kdump_detection	Boolean: true or false	Toggles whether to determine if kdump is running on the host before it is shut down. When set to **true**, the host will not shut down during a kdump process. This is set to **true** when a host has power management enabled, unless disabled by the user.	

The **options** element requires a list of **option** sub-elements. Each **option** requires a **name** and **type** attributes. Certain options are only available for specific fencing types as defined in the **capabilities** collection.

A new host includes an optional **power_management** configuration when **POST**ing to the host resource. The **power_management** configuration is updatable using a **PUT** request.

Example 14.3. An XML representation of a host's power management configuration

```
<host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"
  href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3">
    <name>host1</name>
    ...
    <power_management type="ilo">
        <enabled>true</enabled>
        <address>192.168.1.107</address>
        <username>admin</username>
        <password>p@55w0Rd!</password>
        <options>
            <option name="secure" value="true"/>
            <option name="port" value="54345"/>
            <option name="slot" value="3"/>
        </options>
        <agents>
            <agent id="07f0b9ce-923a-4a96-a532-3c898fa8b6da">
                <type>apc</type>
                <order>1</order>
                <ip>192.168.1.111</ip>
                <user>example</user>
                <password>p@55w0rd!</password>
                <port>9</port>
                <options>
                    <option name="power_wait" value="5"/>
                    <option name="secure" value="false"/>
                </options>
            </agent>
            <agent id="50c71ba2-8495-11e0-b931-e20e458819ed">
                <type>rsa</type>
                <order>2</order>
                <ip>192.168.1.112</ip>
                <user>example</user>
                <password>p@55w0rd!</password>
                <port>9</port>
                <options>
                    <option name="power_wait" value="5"/>
                    <option name="secure" value="false"/>
                </options>
            </agent>
        </agents>
        <automatic_pm_enabled>true</automatic_pm_enabled>
        <kdump_detection>true</kdump_detection>
    </power_management>
    ...
</host>
```

14.5. MEMORY MANAGEMENT ELEMENTS

The API provides two configuration settings for a host's memory management.

Kernel SamePage Merging (KSM) reduces references to memory pages from multiple identical pages to a single page reference. This helps with optimization for memory density. KSM uses the **`ksm`** element.

Example 14.4. Setting KSM memory management

```
PUT /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"
  href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3">
    <ksm>true</ksm>
</host>
```

Transparent Hugepage support expands the size of memory pages beyond the standard 4kB limit. This reduces memory consumption and increases host performance. Transparent Hugepage support uses the **`transparent_hugepages`** element.

Example 14.5. Setting Transparent Hugepage memory management

```
PUT /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"
  href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3">
    <transparent_hugepages>true</transparent_hugepages>
</host>
```

Availability of Transparent Hugepage support is found in the **`capabilities`** collection.

14.6. METHODS

14.6.1. Creating a Host

Creation of a new host requires the **`name`**, **`address`** and **`root_password`** elements.

Example 14.6. Creating a host

```
POST /ovirt-engine/api/hosts HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host>
    <name>host2</name>
    <address>host2.example.com</address>
    <root_password>p@55w0Rd!</root_password>
```

```
</host>
```

The **root_password** element is only included in the client-provided initial representation and is not exposed in the representations returned from subsequent requests.

14.6.2. Updating a Host

The **name**, **description**, **cluster**, **power_management**, **transparent_hugepages** and **ksm** elements are updatable post-creation.

Example 14.7. Updating a host

```
PUT /ovirt-engine/api/hosts/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host>
    <name>host3</name>
</host>
```

14.6.3. Removing a Host

Removal of a host requires a **DELETE** request.

Example 14.8. Removing a host

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-000000000000
HTTP/1.1

HTTP/1.1 204 No Content
```

14.7. SUB-COLLECTIONS

14.7.1. Host Network Attachments Sub-Collection

The **network_attachments** sub-collection represents the network configuration of the host. Each **network_attachment** element represents a network attached to the host and contains the following elements:

Table 14.3. Elements for a host's network attachments

Element	Type	Description	Properties

Element	Type	Description	Properties
network id=	GUID	A reference to the network to which the host is attached.	
host_nic id=	GUID	A reference to the host network interface to which the network is attached.	
ip_address_assignments	complex	The IP configuration of the network. Each **ip_address_assignment** contains **assignment_method** and **ip address= netmask= gateway=** sub-elements.	
properties	complex	Defines custom property keys for the network. Each **property** contains **name** and **value** sub-elements. See Section 14.7.2.3.2, "Network Attachment Custom Properties".	
reported_configurations	complex	A read-only list of configuration properties for the network attachment. The **in_sync** boolean is **false** when the network attachment is out of sync with the logical network definition of the data center. Each **reported_configuration** contains **name**, **expected_value**, **actual_value**, and **in_sync** sub-elements.	
host id=	GUID	A reference to the host.	

Example 14.9. An XML representation of a network attachment on a host

```
<network_attachment href="/ovirt-engine/api/hosts/00000000-0000-0000-
0000-000000000000/networkattachments/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000">
    <network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009" id="00000000-0000-0000-0000-000000000009"/>
    <host_nic href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000" id="00000000-
0000-0000-0000-000000000000"/>
    <ip_address_assignments>
        <ip_address_assignment>
            <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
            <assignment_method>dhcp</assignment_method>
        </ip_address_assignment>
```

```
    </ip_address_assignments>
    <reported_configurations>
        <in_sync>true</in_sync>
        <reported_configuration>
            <name>mtu</name>
            <expected_value>1500</expected_value>
            <actual_value>1500</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>bridged</name>
            <expected_value>true</expected_value>
            <actual_value>true</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>vlan</name>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>boot_protocol</name>
            <expected_value>DHCP</expected_value>
            <actual_value>DHCP</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
    </reported_configurations>
    <host href="/ovirt-engine/api/hosts/f59a29cd-587d-48a3-b72a-
db537eb21957" id="f59a29cd-587d-48a3-b72a-db537eb21957"/>
</network_attachment>
```

When attaching a network to a host, the **network** and **host_nic** elements are required, with either an **id** or a **name**. The **host_nic** ID can refer to either an unused network interface card or a bond.

Example 14.10. Attach a network to a host

```
POST /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments HTTP/1.1
Accept: application/xml
Content-type: application/xml

<network_attachment>
    <network id="00000000-0000-0000-0000-000000000000"/>
    <host_nic id="00000000-0000-0000-0000-000000000000"/>
</network_attachment>
```

The **host_nic**, **ip_address_assignments**, and **properties** elements are updatable post-creation. Changing the **host_nic** ID moves the network to a different network interface card.

Example 14.11. Modifying a host network attachment

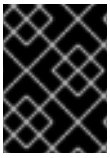
```
PUT /ovirt-engine/api/hosts/00000000-0000-0000-0000-
```

```
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<network_attachment>
    <host_nic id="00000000-0000-0000-0000-000000000000"/>
    <ip_address_assignments>
        <ip_address_assignment>
            <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
     <assignment_method>static</assignment_method>
 </ip_address_assignment>
    </ip_address_assignments>
    <properties>
        <property>
     <name>bridge_opts</name>
     <value>
         forward_delay=1500 group_fwd_mask=0x0 multicast_snooping=1
     </value>
        </property>
    </properties>
</network_attachment>
```

Detach a network from the host with a **DELETE** request on the network attachment.

Example 14.12. Detach a network from a host

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

HTTP/1.1 204 No Content
```



IMPORTANT

Changes to network attachment configuration must be explicitly committed. See Section 14.8.8, “Commit Host Network Configuration Action”.

14.7.2. Host Network Interface Sub-Collection

14.7.2.1. Host Network Interface Sub-Collection

The **nics** sub-collection represents a host's physical network interfaces. Additional information can be retrieved for **GET** requests using the **All-Content: true** header. Each **host_nic** element in the representation acts as a network interface and contains the following elements:

Table 14.4. Elements for a host's network interfaces

Element	Type	Description	Properties
name	string	The name of the host network interface, e.g. **eth0**.	 [a] 
link rel="statistics"	relationship	A link to the **statistics** sub-collection for a host's network interface statistics.	
link rel="labels"	relationship	A link to the **labels** sub-collection for a host's network interface labels.	
link rel="networkattachments"	relationship	A link to the **networkattachments** sub-collection for a host's network interface configuration.	
link rel="master"	relationship	A reference to the master bonded interface, if this is a slave interface.	
host id=	GUID	A reference to the host.	
network id=	GUID	A reference to the network, if any, that the interface is attached.	 [b]
mac address=	string	The MAC address of the interface.	
ip address= netmask= gateway= mtu=	complex	The IP level configuration of the interface.	
mtu	complex	The maximum transmission unit for the interface.	
boot_protocol	enumerated	The protocol for IP address assignment when the host is booting. A list of enumerated values is available in **capabilities**.	
status	enumerated	The link status for the network interface. These states are listed in **host_nic_states** under **capabilities**.	

Element	Type	Description	Properties
vlan id	integer	The VLAN which this interface represents.	
bonding	complex	A list of **options** and **slave** NICs for bonded interfaces.	 [c] 
bridged	Boolean	Defines the bridged network status. Set to **true** for a bridged network and **false** for a bridgeless network.	

[a] Only required when adding bonded interfaces. Other interfaces are read-only and cannot be added.

[b] Only required when adding bonded interfaces. Other interfaces are read-only and cannot be added.

[c] Only required when adding bonded interfaces. Other interfaces are read-only and cannot be added.

Example 14.13. An XML representation of a network interface on a host

```
<host_nic id="00000000-0000-0000-0000-000000000000"
  href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/
  00000000-0000-0000-0000-000000000000">
    <actions>
        <link rel="attach"
      href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/
      00000000-0000-0000-0000-000000000000/attach"/>
        <link rel="detach"
      href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/
      00000000-0000-0000-0000-000000000000/detach"/>
    </actions>
    <name>bond0</name>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000/statistics"
rel="statistics"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000/labels"
rel="labels"/>
    <link href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments" rel="networkattachments"/>
    <host href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000"/>
    <network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000"/>
    <mac address="00:00:00:00:00:00"/>
```

```
    <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
    <boot_protocol>dhcp</boot_protocol>
    <status>
        <state>up</state>
    </status>
    <bonding>
        <options>
            <option name="mode" value="4" type="Dynamic link
aggregation (802.3ad)"/>
            <option name="miimon" value="100"/>
        </options>
        <slaves>
            <host_nic id="00000000-0000-0000-0000-000000000000"/>
            <host_nic id="00000000-0000-0000-0000-000000000000"/>
        </slaves>
    </bonding>
    <mtu>1500</mtu>
    <bridged>true</bridged>
    <custom_configuration>false</custom_configuration>
</host_nic>
```

In the REST API, you can only create bonded interfaces. See Section 14.7.2.2, “Bonded Interfaces”. All other network interfaces contain updatable **network**, **ip** and **boot_protocol** elements.

Modify a network interface with a **PUT** request.

```
PUT /ovirt-engine/api/hosts/00000000-0000-0000-0000-000000000000/nics/
00000000-0000-0000-0000-000000000000 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host_nic>
    <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
    <boot_protocol>static</boot_protocol>
</host_nic>
```

Remove a network interface with a **DELETE** request.

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-000000000000/nics/
00000000-0000-0000-0000-000000000000 HTTP/1.1

HTTP/1.1 204 No Content
```

14.7.2.2. Bonded Interfaces

A bonded interface is represented as a **host_nic** resource containing a **bonding** element.

Table 14.5. Bonded interface properties

Element	Type	Description	Properties
options	complex	A list of **option** elements for a bonded interface. Each **option** contains property **name** and **value** attributes.	[a]
slaves	complex	A list of slave **host_nic id=** elements for a bonded interface.	[b]

[a] Only required when adding bonded interfaces. Other interfaces are read-only and cannot be added.

[b] Only required when adding bonded interfaces. Other interfaces are read-only and cannot be added.

An API user creates a new bond when creating a new **host_nic** (**POST**) or updating a **host_nic** (**PUT**). Use either the **id** or **name** elements to identify the slave **host_nic** elements. When adding a new network interface, the **name** and **network** elements are required. Identify the **network** element with the **id** attribute or **name** element.

Example 14.14. Creating a bonded interface

```
POST /ovirt-engine/api/hosts/00000000-0000-0000-0000-000000000000/nics
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<host_nic>
    <name>bond4</name>
    <network id="00000000-0000-0000-0000-000000000000"/>
    <bonding>
        <slaves>
            <host_nic id="00000000-0000-0000-0000-000000000000"/>
            <host_nic id="00000000-0000-0000-0000-000000000000"/>
        </slaves>
    </bonding>
</host_nic>
```

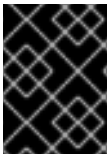
IMPORTANT

bond0, **bond1**, **bond2**, **bond3** and **bond4** are the only valid names for a bonded interface.

Example 14.15. Removing a bonded interface

Remove a bonded interface with a **DELETE** request.

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000 HTTP/1.1

HTTP/1.1 204 No Content
```



IMPORTANT

Changes to bonded interface configuration must be explicitly committed. See Section 14.8.8, “Commit Host Network Configuration Action”.

14.7.2.3. Network Interface Network Attachments

14.7.2.3.1. Network Interface Network Attachments

Each network interface on a host exposes a **network_attachments** sub-collection representing the network interface card's network attachments. Each **network_attachment** represents a network attached to the network interface and contains the following elements:

Table 14.6. Elements for a host network interface's network attachments

Element	Type	Description	Properties
network id=	GUID	A reference to the network to which the interface is attached.	
host_nic id=	GUID	A reference to the host network interface.	
ip_address_assignments	complex	The IP configuration of the network. Each **ip_address_assignment** contains **assignment_method** and **ip address= netmask= gateway=** sub-elements.	
properties	complex	Defines custom property keys for the network. Each **property** contains **name** and **value** sub-elements.	
reported_configurations	complex	A read-only list of configuration properties for the network attachment. The **in_sync** boolean is **false** when the network attachment contains uncommitted network configuration. Each **reported_configuration** contains **name**, **expected_value**, **actual_value**, and **in_sync** sub-elements.	

Example 14.16. An XML representation of a network attachment on a network interface card

```
<network_attachment href="/ovirt-engine/api/hosts/00000000-0000-0000-
0000-000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments/00000000-0000-0000-0000-000000000000"
id="00000000-0000-0000-0000-000000000000">
    <network href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009" id="00000000-0000-0000-0000-000000000009"/>
    <host_nic href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000" id="00000000-
0000-0000-0000-000000000000"/>
    <ip_address_assignments>
        <ip_address_assignment>
            <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
            <assignment_method>static</assignment_method>
        </ip_address_assignment>
    </ip_address_assignments>
    <reported_configurations>
        <in_sync>true</in_sync>
        <reported_configuration>
            <name>mtu</name>
            <expected_value>1500</expected_value>
            <actual_value>1500</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>bridged</name>
            <expected_value>true</expected_value>
            <actual_value>true</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>vlan</name>
            <in_sync>true</in_sync>
        </reported_configuration>
        <reported_configuration>
            <name>boot_protocol</name>
            <expected_value>DHCP</expected_value>
            <actual_value>DHCP</actual_value>
            <in_sync>true</in_sync>
        </reported_configuration>
    </reported_configurations>
</network_attachment>
```

When attaching a network to a network interface card, the **network** element is required, with either an **id** or a **name**.

Example 14.17. Attach a network to a host network interface card

```
POST /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments HTTP/1.1
Accept: application/xml
```

```
Content-type: application/xml

<networkattachment>
    <network id="00000000-0000-0000-0000-000000000000"/>
</networkattachment>
```

The **ip_address_assignments** and **properties** elements are updatable post-creation.

Example 14.18. Modifying a network attachment

```
PUT /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<networkattachment>
    <ip_address_assignments>
        <ip_address_assignment>
            <ip address="XX.XX.XX.XX" netmask="255.255.255.0"
gateway="XX.XX.XX.XX"/>
     <assignment_method>static</assignment_method>
 </ip_address_assignment>
    </ip_address_assignments>
</networkattachment>
```

Detach a network from the network interface card with a **DELETE** request on the network attachment.

Example 14.19. Detach a network from a host network interface card

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-
000000000000/networkattachments/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

HTTP/1.1 204 No Content
```

IMPORTANT

Changes to network attachment configuration must be explicitly committed. See Section 14.8.8, “Commit Host Network Configuration Action”.

14.7.2.3.2. Network Attachment Custom Properties

Custom properties can be applied to network attachments. Each property contains **name** and **value** sub-elements. To amend the custom properties, perform a **PUT** request on a network attachment, or a **POST** request with the **setupnetworks** action.

Table 14.7. Elements for custom bridge options for a host's network interface

Element	Type	Description
name	string	The unique identifier for the property. Bridge options have the set name of **bridge_opts**.
value	string	The bridge options, represented by a valid key and value with the following syntax: [key]=[value]. Separate multiple entries with a whitespace character. The following keys are valid, with the values provided as examples: **forward_delay**=*1500* **gc_timer**=*3765* **group_addr**=*1:80:c2:0:0:0* **group_fwd_mask**=*0x0* **hash_elasticity**=*4* **hash_max**=*512* **hello_time**=*200* **hello_timer**=*70* **max_age**=*2000* **multicast_last_member_count**=*2* **multicast_last_member_interval**=*100* **multicast_membership_interval**=*26000* **multicast_querier**=*0* **multicast_querier_interval**=*25500* **multicast_query_interval**=*13000* **multicast_query_response_interval**=*1000* **multicast_query_use_ifaddr**=*0* **multicast_router**=*1* **multicast_snooping**=*1* **multicast_startup_query_count**=*2* **multicast_startup_query_interval**=*3125*

Example 14.20. An XML representation of a network attachment's properties sub-collection

```
<network_attachment>
  ...
  <properties>
    <property>
      <name>bridge_opts</name>
      <value>
        forward_delay=1500 group_fwd_mask=0x0 multicast_snooping=1
      </value>
    </property>
  </properties>
  ...
</network_attachment>
```

14.7.2.4. Network Interface Labels

You can attach labels to a host network interface card to automate the association of that network interface card with logical networks to which the same label has been attached.

-

Example 14.21. Attaching a label to a network interface card

```
POST /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000/labels HTTP/1.1
Accept: application/xml
Content-type: application/xml

<label id="Label_001" />
```

Removal of a label from a physical host network interface card requires a **DELETE** request.

Example 14.22. Removing a label from a network interface card

```
DELETE /ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/00000000-0000-0000-0000-000000000000/labels/00000000-
0000-0000-0000-000000000000 HTTP/1.1

HTTP/1.1 204 No Content
```

14.7.2.5. Network Interface Statistics

Each host's network interface exposes a **statistics** sub-collection for a host's network interface statistics. Each **statistic** contains the following elements:

Table 14.8. Elements for a host's network interface statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
host_nic id=	relationship	A relationship to the containing **host_nic** resource.

The following table lists the statistic types for network interfaces on hosts.

Table 14.9. Host NIC statistic types

Name	Description
data.current.rx	The rate in bytes per second of data received.
data.current.tx	The rate in bytes per second of data transmitted.
data.total.rx	Total received data.
data.total.tx	Total transmitted data.
errors.total.rx	Total errors from receiving data.
errors.total.tx	Total errors from transmitting data.

Example 14.23. An XML representation of a host's network interface statistics sub-collection

```
<statistics>
    <statistic id="00000000-0000-0000-0000-000000000000"
      href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/
      00000000-0000-0000-0000-000000000000/statistics/
      00000000-0000-0000-0000-000000000000">
        <name>data.current.rx</name>
        <description>Receive data rate</description>
        <values type="DECIMAL">
            <value>
                <datum>0</datum>
            </value>
        </values>
        <type>GAUGE</type>
        <unit>BYTES_PER_SECOND</unit>
        <host_nic id="00000000-0000-0000-0000-000000000000"
          href="/ovirt-engine/api/hosts/00000000-0000-0000-0000-
000000000000/nics/
          00000000-0000-0000-0000-000000000000"/>
    </statistic>
    ...
</statistics>
```

NOTE

This **statistics** sub-collection is read-only.

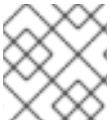
14.7.3. Storage Sub-Collection

The **storage** sub-collection provides a list of the iSCSI and FCP storage representations available on the host. This storage is used to create storage domains.

Each **storage** representation in the sub-collection represents a SCSI LUN.

Example 14.24. An XML representation of the storage sub-collection on a host

```
<host_storage>
    <storage id="82fb123b-321e-40a1-9889-95dcd2654463"
      href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/storage/
      82fb123b-321e-40a1-9889-95dcd2654463">
        <name>LUN0</name>
        <type>iscsi</type>
        <logical_unit id="LUN0">
            <address>mysan.example.com</address>
            <target>iqn.2009-08.com.example:mysan.foobar</target>
        </logical_unit>
    </storage>
</host_storage>
```



NOTE

The **host_storage** collection is read-only.

IMPORTANT

The API as documented in this section is experimental and subject to change. It is not covered by the backwards compatibility statement.

14.7.4. Host NUMA Nodes Sub-Collection

14.7.4.1. NUMA Nodes Sub-Collection

The **numanodes** sub-collection represents the host's NUMA topology. Each **host_numa_node** element in the sub-collection represents a NUMA node.

Example 14.25. An XML representation of the numanodes sub-collection on a host

```
<host_numa_nodes>
    <host_numa_node href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-
b750-a87863736cc2/numanodes/91d8537c-699e-460b-9a70-285f651e7d68"
id="91d8537c-699e-460b-9a70-285f651e7d68">
        <link href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-b750-
a87863736cc2/numanodes/91d8537c-699e-460b-9a70-285f651e7d68/statistics"
rel="statistics"/>
        <host href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-b750-
a87863736cc2" id="f6735fa9-4ee5-47ce-b750-a87863736cc2"/>
        <index>0</index>
        <memory>8157</memory>
        <cpu>
            <cores>
                <core index="0"/>
                <core index="2"/>
                <core index="4"/>
```

```
                <core index="6"/>
            </cores>
        </cpu>
        <node_distance>10 16</node_distance>
    </host_numa_node>
    <host_numa_node href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-
b750-a87863736cc2/numanodes/4b18926e-6faf-43f5-9fc2-0503f1531562"
id="4b18926e-6faf-43f5-9fc2-0503f1531562">
        <link href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-b750-
a87863736cc2/numanodes/4b18926e-6faf-43f5-9fc2-0503f1531562/statistics"
rel="statistics"/>
        <host href="/ovirt-engine/api/hosts/f6735fa9-4ee5-47ce-b750-
a87863736cc2" id="f6735fa9-4ee5-47ce-b750-a87863736cc2"/>
        <index>2</index>
        <memory>8175</memory>
        <cpu>
            <cores>
                <core index="1"/>
                <core index="3"/>
                <core index="5"/>
                <core index="7"/>
            </cores>
        </cpu>
        <node_distance>16 10</node_distance>
    </host_numa_node>
</host_numa_nodes>
```

NOTE

The **host_numa_nodes** sub-collection is read-only.

14.7.4.2. NUMA Node Statistics

Each host NUMA node exposes a **statistics** sub-collection for NUMA node statistics. Each **statistic** contains the following elements:

Table 14.10. Elements for a host's NUMA node statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.

Element	Type	Description
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
host_numa_node id=	relationship	A relationship to the containing **numanode** resource.

The following table lists the statistic types for host NUMA nodes.

Table 14.11. Host NUMA node statistics

Name	Description
memory.total	Total memory in bytes on the NUMA node.
memory.used	Memory in bytes used on the NUMA node.
memory.free	Memory in bytes free on the NUMA node.
cpu.current.user	Percentage of CPU usage for users.
cpu.current.system	Percentage of CPU usage for the system.
cpu.current.idle	Percentage of idle CPU usage.

Example 14.26. An XML representation of the host NUMA node's statistics sub-collection

```
<statistics>
    <statistic href="/ovirt-engine/api/hosts/f6745fa9-4ee5-47ce-b750-
a87863736cc2/numanodes/91d8537c-689e-460b-9a70-
285f651e7d68/statistics/7816602b-c05c-3dc7-a4da-3769f7ad8896"
id="7816602b-c05c-3dc7-a4da-3769f7ad8896">
        <name>memory.total</name>
        <description>Total memory</description>
        <values type="INTEGER">
            <value>
                <datum>8157</datum>
            </value>
        </values>
        <type>GAUGE</type>
        <unit>BYTES</unit>
        <host_numa_node href="/ovirt-engine/api/hosts/f6745fa9-4ee5-
47ce-b750-a87863736cc2/numanodes/91d8537c-689e-460b-9a70-285f651e7d68"
id="91d8537c-689e-460b-9a70-285f651e7d68"/>
    </statistic>
    ...
</statistics>
```

NOTE

A host NUMA node's **statistics** sub-collection is read-only.

14.7.5. Host Statistics Sub-Collection

14.7.5.1. Host Statistics Sub-Collection

Each host resource exposes a **statistics** sub-collection for host-specific statistics. Each **statistic** contains the following elements:

Table 14.12. Elements for host statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
host id=	relationship	A relationship to the containing **host** resource.

The following table lists the statistic types for hosts.

Table 14.13. Host statistic types

Name	Description
memory.total	Total memory in bytes on the host.
memory.used	Memory in bytes used on the host.
memory.free	Memory in bytes free on the host.
memory.shared	Memory in bytes shared on the host.

Name	Description
memory.buffers	I/O buffers in bytes.
memory.cached	OS caches in bytes.
swap.total	Total swap memory in bytes on the host.
swap.free	Swap memory in bytes free on the host.
swap.used	Swap memory in bytes used on the host.
swap.cached	Swap memory in bytes also cached in host's memory.
ksm.cpu.current	Percentage of CPU usage for Kernel SamePage Merging.
cpu.current.user	Percentage of CPU usage for users.
cpu.current.system	Percentage of CPU usage for system.
cpu.current.idle	Percentage of idle CPU usage.
cpu.load.avg.5m	CPU load average per five minutes.

Example 14.27. An XML representation of the host's statistics sub-collection

```
<statistics>
    <statistic id="4ae97794-f56d-3f05-a9e7-8798887cd1ac"
      href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/
      statistics/4ae97794-f56d-3f05-a9e7-8798887cd1ac">
        <name>memory.total</name>
        <description>Total memory</description>
        <unit>BYTES</unit>
        <type>GUAGE</type>
        <values type="INTEGER">
            <value>
                <datum>3983540224<datum>
            </value>
        </values>
        <host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"
          href="/ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3"/>
    </statistic>
    ...
</statistics>
```

NOTE

A host's **statistics** sub-collection is read-only.

14.8. ACTIONS

14.8.1. Install VDSM Action

Install VDSM and related software on the host.

Example 14.28. Action to install VDSM on a virtualization host

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/install HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <root_password>p@55w0Rd!</root_password>
</action>
```

14.8.2. Activate Host Action

Activate the host for use, such as running virtual machines.

Example 14.29. Action to activate a host

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/activate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

14.8.3. Host Network Setup Action

Configure multiple network settings on a host. The **setupnetworks** action can be used for complex network configuration such as moving a network from one network interface to another.

Example 14.30. Action to edit host network configuration

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/setupnetworks HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <modified_network_attachments>
        <network_attachment id="41561e1c-c653-4b45-b9c9-126630e8e3b9">
            <host_nic id="857a46d3-5f64-68bd-f456-c70de5b2d569"/>
```

```
        </network_attachment<
        <network_attachment id="3c3f442f-948b-4cdc-9a48-89bb0593cfbd">
            <network id="00000000-0000-0000-0000-000000000010"/>
            <ip address="10.35.1.247" netmask="255.255.254.0"
gateway="10.35.1.254"/>
     <properties>
  <property>
      <name>bridge_opts</name>
      <value>
   forward_delay=1500 group_fwd_mask=0x0 multicast_snooping=1
      </value>
  </property>
     </properties>
        </network_attachment>
    </modified_network_attachments>
    <synchronized_network_attachments>
        <network_attachment id="3c3f442f-948b-4cdc-9a48-89bb0593cfbd">
    </synchronized_network_attachments>
    <removed_network_attachments>
        <network_attachment id="7f456dae-c57f-35d5-55a4-20b74dc53af9">
    </removed_network_attachments>
    <modified_bonds>
        <host_nic id="a56b212d-2bc4-4120-9136-53be6cacb39a">
     <bonding>
  <slaves>
      <host_nic id="75ac21f7-4aa3-405a-a022-341e5f525b85">
      <host_nic id="f3dda04c-1233-41af-a111-74327b876487">
  </slaves>
     </bonding>
        </host_nic>
    </modified_bonds>
    <removed_bonds>
        <host_nic id="36ab5c7f-647a-bc64-f5e7-ba5d74f8e4ba">
    </removed_bonds>
    <modified_labels>
        <label id="Label002">
     <host_nic id="857a46d3-5f64-68bd-f456-c70de5b2d569"/>
        </label>
        <label>
     <host_nic id="a56b212d-2bc4-4120-9136-53be6cacb39a"/>
     <label id="Label003/>
        </label>
    </modified_labels>
    <removed_labels>
        <label id="Label001">
    </removed_labels>
    <checkConnectivity>true</checkConnectivity>
    <connectivityTimeout>60</connectivityTimeout>
</action>
```

This action updates all specified host network resources with standard NIC elements. The request includes additional elements specified in the following table.

Table 14.14. Elements for multiple host network interface setup

Element	Type	Description
modified_bonds	complex	Creates or updates bonds. Each **host_nic** element contains standard **bonding** elements. See Section 14.7.2.2, "Bonded Interfaces".
removed_bonds	complex	An ID list of bonds to remove.
modified_network_attachments	complex	Adds or updates network attachments on the host. Each **network_attachment** element contains standard host **network_attachment** elements. See Section 14.7.1, "Host Network Attachments Sub-Collection". Changing the **host_nic** ID moves the network to a different network interface card.
synchronized_network_attachments	complex	An ID list of out-of-sync network attachments to synchronize with the logical network definition of the data center.
removed_network_attachments	complex	An ID list of network attachments to remove.
modified_labels	complex	Creates or modifies labels. Each **label** element contains a **label id** (when creating a label) and a **host_nic** identified by a name or ID. Changing the **host_nic** ID moves the label to a different network interface card.
removed_labels	complex	An ID list of labels to remove.
checkConnectivity	Boolean	Set to **true** to verify connectivity between the host and the Red Hat Virtualization Manager. If the connectivity is lost, Red Hat Virtualization Manager reverts the settings.
connectivityTimeout	integer	Defines the timeout for loss of connectivity.

14.8.4. Fence Host Action

An API user controls a host's power management device with the **fence** action. The **capabilities** lists available **fence_type** options.

Example 14.31. Action to fence a host

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/fence
Accept: application/xml
Content-Type: application/xml
```

```
<action>
    <fence_type>start</fence_type>
</action>
```

14.8.5. Deactivate Host Action

Deactivate the host to perform maintenance tasks.

Example 14.32. Action to deactivate a host

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/deactivate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

14.8.6. Host iSCSI Login Action

The **`iscsilogin`** action enables a host to login to an iSCSI target. Logging into a target makes the contained LUNs available in the **`host_storage`** collection.

Example 14.33. Action to enable a host to login to iSCSI target

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/iscsilogin HTTP/1.1
Accept: application/xml
Content-Type: application/xml


<action>
    <iscsi>
        <address>mysan.example.com</address>
        <target>iqn.2009-08.com.example:mysan.foobar</target>
        <username>jimmy</username>
        <password>s3kr37</password>
    </iscsi>
</action>
```

14.8.7. Host iSCSI Discover Action

The **`iscsidiscover`** action enables an iSCSI portal to be queried for its list of targets.

Example 14.34. Action to query a list of targets for iSCSI portal

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/iscsidiscover HTTP/1.1
Accept: application/xml
```

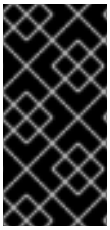
```
Content-Type: application/xml

<action>
    <iscsi>
        <address>mysan.example.com</address>
        <port>3260</port>
    </iscsi>
</action>
```

14.8.8. Commit Host Network Configuration Action

An API user commits the network configuration to persist a host network interface attachment or detachment, or persist the creation and deletion of a bonded interface.

Example 14.35. Action to commit network configuration

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/commitnetconfig HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```



IMPORTANT

Networking configuration is only committed after the Manager has established that host connectivity is not lost as a result of the configuration changes. If host connectivity is lost, the host requires a reboot and automatically reverts to the previous networking configuration.

14.8.9. Setting SPM

Manually set a host as the Storage Pool Manager (SPM).

Example 14.36. Action to Set Host as SPM

```
POST /ovirt-engine/api/hosts/2ab5e1da-b726-4274-bbf7-
0a42b16a0fc3/forceselectspm HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

CHAPTER 15. VIRTUAL MACHINES

15.1. VIRTUAL MACHINE ELEMENTS

The **vms** collection provides information about virtual machines in a Red Hat Virtualization environment. An API user accesses this information through the **rel="vms"** link obtained from the entry point URI.

Additional information can be retrieved for **GET** requests using the **All-Content: true** header.

The following table shows specific elements contained in a virtual machine resource representation.

Table 15.1. Virtual machine elements

Element	Type	Description	Properties
link rel="applications"	relationship	A link to the **applications** sub-collection for virtual machine resources, which shows the applications installed on the virtual machine.	
link rel="disks"	relationship	A link to the **disks** sub-collection for virtual machine resources.	
link rel="nics"	relationship	A link to the **nics** sub-collection for virtual machine resources.	
link rel="numanodes"	relationship	A link to the **numanodes** sub-collection for virtual machine resources.	
link rel="cdroms"	relationship	A link to the **cdroms** sub-collection for virtual machine resources.	
link rel="snapshots"	relationship	A link to the **snapshots** sub-collection for virtual machine resources.	
link rel="tags"	relationship	A link to the **tags** sub-collection for virtual machine resources.	
link rel="permissions"	relationship	A link to the **permissions** sub-collection for virtual machine permissions.	
link rel="statistics"	relationship	A link to the **statistics** sub-collection for virtual machine resources.	
link rel="reporteddevices"	relationship	A link to the **reporteddevices** sub-collection for virtual machine resources.	

Element	Type	Description	Properties
link rel="watchdogs"	relationship	A link to the **watchdogs** sub-collection for virtual machine resources.	
link rel="sessions"	relationship	A link to the **sessions** sub-collection for virtual machine resources.	
type	enumerated	The virtual machine type. A list of enumerated values are available in **capabilities**.	
status	See below	The virtual machine status.	
memory	integer	The amount of memory allocated to the guest in bytes.	
cpu	complex	Defines CPU details for the virtual machine. The **topology** sub-element sets number of logical **sockets** available to the guest and the number of **cores** per socket. The total cores available to the virtual machine equals the number of sockets multiplied by the cores per socket. The **cputune** sub-element maps virtual CPUs to physical host CPUs using a series of **vcpupin** elements. Each **vcpupin** elements contains a virtual CPU attribute (**vcpu**) and an attribute to define which physical to use (**cpuset**). Set the **cpuset** to either a single CPU (**cpuset="0"**), multiple CPUs (**cpuset="0,2"**), a CPU range (**cpuset="0-3"**) or a CPU range with exclusion (**cpuset="0-3,^2"**). The **cpu_mode** sub-element defines how closely the virtual CPU relates to the host CPU. It has three values: **custom** is the default if no mode is given, **host_model** copies the host CPU as best as libvirt can understand, and **host_passthrough** passes all aspects of the host to the guest, even those that libvirt does not recognize. However, **host_passthrough** will prevent migration of that virtual machine.	
os type=	string, e.g. **RHEL5** or **WindowsXP**	The guest operating system type.	
os boot dev=	enumerated	A list of boot devices described by a **dev** attribute on a **boot** element. A list of enumerated values are available in **capabilities**.	

Element	Type	Description	Properties
os kernel	string	A path to a kernel image the virtual machine is configured to boot. This option supports booting a Linux kernel directly rather than through the BIOS bootloader.	
os initrd	string	A path to an initrd image to be used with the previously specified kernel. This option supports booting a Linux kernel directly rather than through the BIOS bootloader.	
os cmdline	string	A kernel command line parameter string to be used with the defined kernel. This option supports booting a Linux kernel directly rather than through the BIOS bootloader.	
high_availability	complex	Set **enabled** to **true** if the virtual machine should be automatically restarted if the virtual machine or its host crashes. A **priority** element controls the order in which virtual machines are re-started.	
display	complex	The display **type** (either **vnc** or **spice**), port, and the number of **monitors**. The **allow_reconnect** Boolean value specifies if a client can reconnect to the machine via display. The **smartcard_enabled** sub-element is a Boolean (**true** or **false**) to specify if a Smartcard attached to a client is passed through to a virtual machine. The default is **false**.	
cluster id=	GUID	A reference to the virtual machine's host cluster.	
template id=	GUID	A reference to the template on which this virtual machine is based.	 
domain id=	GUID	A reference to the virtual machine's domain.	
start_time	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The date and time at which this virtual machine was started.	

Element	Type	Description	Properties
stop_time	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The date and time at which this virtual machine was stopped.	
creation_time	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The date and time at which this virtual machine was created.	
origin	One of **rhev**, **ovirt**, **vmware** or **xen**	The system from which this virtual machine originated.	
stateless	Boolean: true or false	**true** if the virtual machine is stateless. A stateless virtual machine contains a snapshot of its disk image taken at boot and deleted at shutdown. This means state changes do not persist after a reboot.	
delete_protected	Boolean: true or false	If set to **true**, the virtual machine cannot be deleted.	
sso	string	A reference to the method of single sign-on for the virtual machine. Includes a **method** element with an **ip** attribute.	
placement_policy	complex	Sets the placement policy for virtual machine migration. Requires a default **host=** and an **affinity** (one of **migratable**, **user_migratable** or **pinned**). Leave the **host** element empty to set no preferred host. Use multiple **host** elements to specify a subset of preferred hosts within a cluster.	
memory_policy	complex	Sets the memory policy for virtual machines. Defines the minimum amount of **guaranteed** memory on a host in order for the virtual machine to run.	
quota id=	GUID	Sets a quota for the virtual machine.	
custom_properties	complex	A set of user-defined environment variable passed as parameters to custom scripts. Each **custom_property** contains **name** and **value** attributes. A list of enumerated values are available in **capabilities**.	

Element	Type	Description	Properties
usb	complex	Defines the USB policy for a virtual machine. Requires an **enabled** element set to a Boolean value and a **type** element set to either **native** or **legacy**.  **IMPORTANT** The Legacy USB option has been deprecated and will be removed in Red Hat Virtualization 4.1.	
migration_downtime	integer	Represents the maximum number of milliseconds the virtual machine can be down during live migration. A value of 0 means that the VDSM default will be used.	
cpu_profile id=	GUID	A reference to the virtual machine's cpu profile.	
next_run_configuration	Boolean: true or false	**true** if changes to the virtual machine's configuration will be applied when the virtual machine is next restarted.	
numa_tune_mode	string	Reference to the mode of memory allocation (**interleave**, **strict**, or **preferred**) of the host NUMA node.	
guest_info	complex	A reference to the guest client information. Includes an **ip** element with an **address=** attribute.	
vmpool	complex	A reference to the virtual machine pool. This element only appears for virtual machines part of a pool.	
timezone	tz database format: **Area/Location**	The Sysprep timezone setting for a Windows virtual machine.	
domain	complex	The Sysprep domain setting for a Windows virtual machine. Requires a **name** from the **domains** collection.	
initialization	complex	Defines a list of values applied to the virtual machine on boot using Cloud-Init for Linux-based virtual machines, or Sysprep for Windows-based virtual machines. **Cloud-Init**	

Element	Type	Description	Properties
		• **host_name**: The host name of the virtual machine. • **timezone**: The time zone for the virtual machine. • **user_name**: The user name for the virtual machine. • **root_password**: The password for the user, or root password if no user is specified. • **authorized_ssh_keys**: SSH keys to be added to the authorized keys file of the virtual machine. You can enter multiple SSH keys by separating each SSH key with a line break. • **regenerate_ssh_keys**: Whether to regenerate SSH key for the virtual machine. Possible values are **true** or **false**. • **dns_servers**: A space-separated list of DNS servers. • **dns_search**: A space-separated list of DNS search domains. • **nic_configurations**: Defines a network interface controller for the virtual machine. Network interface controllers are defined as **nic_configuration** objects under this collection that each specify the **name**, **ip**, **boot_protocol**, and **on_boot**. • **custom_script**: A custom script to run on the virtual machine when it starts. **Sysprep** • **host_name**: The host name of the virtual machine. • **domain**: The domain of which the virtual machine is a member. • **authorized_ssh_keys**: SSH keys to be added to the authorized keys file of the virtual machine. You can enter multiple SSH keys by separating each SSH key with a line break. • **regenerate_ssh_keys**: Whether to regenerate SSH key for the virtual machine. Possible values are **true** or **false**. • **timezone**: The time zone for the virtual machine. • **root_password**: The password for the admin user of the virtual machine.	

Element	Type	Description	Properties
		• **custom_script**: A custom script to run on the virtual machine when it starts. • **input_locale**: The locale for user input. • **ui_language**: The language used for user interface elements such as buttons and menus. • **system_locale**: The locale for the overall system. • **user_locale**: The locale for users. • **active_directory_ou**: The organizational unit in the Active Directory domain to which the virtual machine belongs. • **org_name**: The name of the organization to which the virtual machine belongs.	
payloads	complex	Defines a set of **payload** elements to deliver content to a virtual machine upon boot. Each **payload** requires a **type** attribute, either **cdrom** or **floppy**, and a set of **file** elements. Within each **file** element is a **name** element that specifies the name and location of the file, and a **content** element that defines the content to deliver to the file. The **payloads** element is used by the **cloud-init** feature. When **cloud-init** is used to configure a virtual machine, a payload is automatically created with the **type** attribute set to **cd-rom** and two **file** sub-elements, **openstack/latest/meta_data.json** and **openstack/latest/user_data**, which pass configuration parameters to the virtual machine.	

The **status** contains one of the following enumerative values: **unassigned**, **down**, **up**, **powering_up**, **powered_down**, **paused**, **migrating_from**, **migrating_to**, **unknown**, **not_responding**, **wait_for_launch**, **reboot_in_progress**, **saving_state**, **restoring_state**, **suspended**, **image_illegal**, **image_locked** or **powering_down**. These states are listed in **vm_states** under **capabilities**.

15.2. XML REPRESENTATION OF A VIRTUAL MACHINE

Example 15.1. An XML representation of a virtual machine

```
<vm id="70b4d9a7-4f73-4def-89ca-24fc5f60e01a"
  href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-24fc5f60e01a">
    <actions>
        <link rel="move"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
```

```
24fc5f60e01a/move"/>
        <link rel="ticket"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/ticket"/>
        <link rel="reboot"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/reboot"/>
        <link rel="undo_snapshot"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/undo_snapshot"/>
        <link rel="commit_snapshot"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/commit_snapshot"/>
        <link rel="preview_snapshot"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/preview_snapshot"/>
        <link rel="logon"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/logon"/>
        <link rel="cancelmigration"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/cancelmigration"/>
        <link rel="maintenance"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/maintenance"/>
        <link rel="clone"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/clone"/>
        <link rel="migrate"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/migrate"/>
        <link rel="detach"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/detach"/>
        <link rel="export"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/export"/>
        <link rel="shutdown"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/shutdown"/>
        <link rel="start"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/start"/>
        <link rel="stop"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/stop"/>
        <link rel="suspend"
   href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/suspend"/>
    </actions>
    <name>VM_01</name>
    <description>Testing Virtual Machine</description>
    <link rel="applications"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/applications"/>
    <link rel="disks"
```

```
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/disks"/>
    <link rel="nics"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/nics"/>
    <link rel="numanodes"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/numanodes"/>
    <link rel="cdroms"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/cdroms"/>
    <link rel="snapshots"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/snapshots"/>
    <link rel="tags"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/tags"/>
    <link rel="permissions"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/permissions"/>
    <link rel="statistics"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/statistics"/>
    <link rel="reporteddevices"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/reporteddevices"/>
    <link rel="watchdogs"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/watchdogs"/>
    <link rel="sessions"
      href="/ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-
24fc5f60e01a/sessions"/>
    <type>server</type>
    <status>
        <state>down</state>
    </status>
    <memory>1073741824</memory>
    <cpu>
        <topology sockets="1" cores="1"/>
        <architecture>X86_64</architecture>
    </cpu>
    <cpu_shares>0</cpu_shares>
    <bios>
        <boot_menu>
            <enabled>false</enabled>
        </boot_menu>
    </bios>
    <os type="other">
        <boot dev="hd"/>
    </os>
    <high_availability>
        <enabled>false</enabled>
        <priority>1</priority>
    </high_availability>
    <display>
        <type>spice</type>
```

```
        <monitors>1</monitors>
        <single_qxl_pci>false</single_qxl_pci>
        <allow_override>true</allow_override>
        <smartcard_enabled>false</smartcard_enabled>
        <file_transfer_enabled>true</file_transfer_enabled>
        <copy_paste_enabled>true</copy_paste_enabled>
    </display>
    <cluster href="/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb" id="00000001-0001-0001-0001-0000000002fb"/>
    <template href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000"/>
    <stop_time>2014-12-03T14:25:45.588+10:00</stop_time>
    <creation_time>2014-12-03T14:25:45.535+10:00</creation_time>
    <origin>ovirt</origin>
    <stateless>false</stateless>
    <delete_protected>false</delete_protected>
    <sso>
        <methods>
            <method id="GUEST_AGENT"/>
        </methods>
    </sso>
    <timezone>Etc/GMT</timezone>
    <placement_policy>
        <affinity>migratable</affinity>
    </placement_policy>
    <memory_policy>
        <guaranteed>1073741824</guaranteed>
    </memory_policy>
    <usb>
        <enabled>false</enabled>
    </usb>
    <migration_downtime>-1</migration_downtime>
    <cpu_profile href="/ovirt-engine/api/cpuprofiles/0000001a-001a-001a-
001a-0000000002e3" id="0000001a-001a-001a-001a-0000000002e3"/>
    <next_run_configuration_exists>false</next_run_configuration_exists>
    <numa_tune_mode>interleave</numa_tune_mode>
</vm>
```

15.3. XML REPRESENTATION OF ADDITIONAL OVF DATA FOR A VIRTUAL MACHINE

Use a **GET** request for a virtual machine with the **All-Content: true** header to include additional OVF data with the representation of the virtual machine.

The **Accept** header defaults to **application/xml** if left blank, and the data is represented with HTML entities so as not to interfere with the XML tags. Specifying the **Accept: application/json** header will return the data in standard XML tagging. This example representation has been formatted from its standard block format to improve legibility.

Example 15.2. XML representation of additional ovf data for a virtual machine

```
GET /ovirt-engine/api/vms/70b4d9a7-4f73-4def-89ca-24fc5f60e01a HTTP/1.1
All-Content: true
```

```
<?xml version='1.0' encoding='UTF-8'?>
<ovf:Envelope xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/1/"
  xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/CIM_ResourceAllocationSettingData"
  xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/CIM_VirtualSystemSettingData"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  ovf:version="3.5.0.0">
  <References/>
  <Section xsi:type="ovf:NetworkSection_Type">
    <Info>List of networks</Info>
    <Network ovf:name="Network 1"/>
  </Section>
  <Section xsi:type="ovf:DiskSection_Type">
    <Info>List of Virtual Disks</Info>
  </Section>
  <Content ovf:id="out" xsi:type="ovf:VirtualSystem_Type">
    <CreationDate>2014/12/03 04:25:45</CreationDate>
    <ExportDate>2015/02/09 14:12:24</ExportDate>
    <DeleteProtected>false</DeleteProtected>
    <SsoMethod>guest_agent</SsoMethod>
    <IsSmartcardEnabled>false</IsSmartcardEnabled>
    <TimeZone>Etc/GMT</TimeZone>
    <default_boot_sequence>0</default_boot_sequence>
    <Generation>1</Generation>
    <VmType>1</VmType>
    <MinAllocatedMem>1024</MinAllocatedMem>
    <IsStateless>false</IsStateless>
    <IsRunAndPause>false</IsRunAndPause>
    <AutoStartup>false</AutoStartup>
    <Priority>1</Priority>
    <CreatedByUserId>fdfc627c-d875-11e0-90f0-
83df133b58cc</CreatedByUserId>
    <IsBootMenuEnabled>false</IsBootMenuEnabled>
    <IsSpiceFileTransferEnabled>true</IsSpiceFileTransferEnabled>
    <IsSpiceCopyPasteEnabled>true</IsSpiceCopyPasteEnabled>
    <Name>VM_export</Name>
    <TemplateId>00000000-0000-0000-0000-000000000000</TemplateId>
    <TemplateName>Blank</TemplateName>
    <IsInitilized>false</IsInitilized>
    <Origin>3</Origin>
    <DefaultDisplayType>1</DefaultDisplayType>
    <TrustedService>false</TrustedService>
    <OriginalTemplateId>00000000-0000-0000-0000-
000000000000</OriginalTemplateId>
    <OriginalTemplateName>Blank</OriginalTemplateName>
    <UseLatestVersion>false</UseLatestVersion>
    <Section ovf:id="70b4d9a7-4f73-4def-89ca-24fc5f60e01a"
      ovf:required="false"
      xsi:type="ovf:OperatingSystemSection_Type">
      <Info>Guest Operating System</Info>
      <Description>other</Description>
    </Section>
    <Section xsi:type="ovf:VirtualHardwareSection_Type">
      <Info>1 CPU, 1024 Memeory</Info>
```

```
      <System>
        <vssd:VirtualSystemType>ENGINE 3.5.0.0</vssd:VirtualSystemType>
      </System>
      <Item>
        <rasd:Caption>1 virtual cpu</rasd:Caption>
        <rasd:Description>Number of virtual CPU</rasd:Description>
        <rasd:InstanceId>1</rasd:InstanceId>
        <rasd:ResourceType>3</rasd:ResourceType>
        <rasd:num_of_sockets>1</rasd:num_of_sockets>
        <rasd:cpu_per_socket>1</rasd:cpu_per_socket>
      </Item>
      <Item>
        <rasd:Caption>1024 MB of memory</rasd:Caption>
        <rasd:Description>Memory Size</rasd:Description>
        <rasd:InstanceId>2</rasd:InstanceId>
        <rasd:ResourceType>4</rasd:ResourceType>
        <rasd:AllocationUnits>MegaBytes</rasd:AllocationUnits>
        <rasd:VirtualQuantity>1024</rasd:VirtualQuantity>
      </Item>
      <Item>
        <rasd:Caption>USB Controller</rasd:Caption>
        <rasd:InstanceId>3</rasd:InstanceId>
        <rasd:ResourceType>23</rasd:ResourceType>
        <rasd:UsbPolicy>DISABLED</rasd:UsbPolicy>
      </Item>
    </Section>
  </Content>
</ovf:Envelope>
```

15.4. JSON REPRESENTATION OF A VIRTUAL MACHINE

Example 15.3. A JSON representation of a virtual machine

```
{
  "type" : "server",
  "status" : {
    "state" : "down"
  },
  "stop_reason" : "",
  "memory" : 1073741824,
  "cpu" : {
    "topology" : {
      "sockets" : "1",
      "cores" : "1"
    },
    "architecture" : "X86_64"
  },
  "cpu_shares" : "0",
  "bios" : {
    "boot_menu" : {
      "enabled" : "false"
    }
  },
  "os" : {
```

```
    "boot" : [ {
      "dev" : "hd"
    } ],
    "type" : "other"
  },
  "high_availability" : {
    "enabled" : "false",
    "priority" : "1"
  },
  "display" : {
    "type" : "spice",
    "monitors" : "1",
    "single_qxl_pci" : "false",
    "allow_override" : "false",
    "smartcard_enabled" : "false",
    "file_transfer_enabled" : "true",
    "copy_paste_enabled" : "true"
  },
  "cluster" : {
    "href" : "/ovirt-engine/api/clusters/00000001-0001-0001-0001-
0000000002fb",
    "id" : "00000001-0001-0001-0001-0000000002fb"
  },
  "template" : {
    "href" : "/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000",
    "id" : "00000000-0000-0000-0000-000000000000"
  },
  "stop_time" : 1423550982110,
  "creation_time" : 1423490033647,
  "origin" : "ovirt",
  "stateless" : "false",
  "delete_protected" : "false",
  "sso" : {
    "methods" : {
      "method" : [ {
        "id" : "GUEST_AGENT"
      } ]
    }
  },
  "timezone" : "Etc/GMT",
  "initialization" : {
    "regenerate_ssh_keys" : "false",
    "nic_configurations" : { }
  },
  "placement_policy" : {
    "affinity" : "migratable"
  },
  "memory_policy" : {
    "guaranteed" : 1073741824,
    "ballooning" : "true"
  },
  "usb" : {
    "enabled" : "false"
  },
  "migration_downtime" : "-1",
```

```
  "cpu_profile" : {
    "href" : "/ovirt-engine/api/cpuprofiles/0000001a-001a-001a-001a-
0000000002e3",
    "id" : "0000001a-001a-001a-001a-0000000002e3"
  },
  "next_run_configuration_exists" : "false",
  "numa_tune_mode" : "interleave",
  "actions" : {
    "link" : [ {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/ticket",
      "rel" : "ticket"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/move",
      "rel" : "move"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/clone",
      "rel" : "clone"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/commit_snapshot",
      "rel" : "commit_snapshot"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/preview_snapshot",
      "rel" : "preview_snapshot"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/logon",
      "rel" : "logon"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/cancelmigration",
      "rel" : "cancelmigration"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/maintenance",
      "rel" : "maintenance"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/reboot",
      "rel" : "reboot"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/undo_snapshot",
      "rel" : "undo_snapshot"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/migrate",
      "rel" : "migrate"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/detach",
      "rel" : "detach"
```

```
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/export",
      "rel" : "export"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/shutdown",
      "rel" : "shutdown"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/start",
      "rel" : "start"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/stop",
      "rel" : "stop"
    }, {
      "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/suspend",
      "rel" : "suspend"
    } ]
  },
  "name" : "VM_01",
  "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-973a44b2562e",
  "id" : "42ec2621-7ad6-4ca2-bd68-973a44b2562e",
  "link" : [ {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/applications",
    "rel" : "applications"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/disks",
    "rel" : "disks"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/nics",
    "rel" : "nics"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/numanodes",
    "rel" : "numanodes"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/cdroms",
    "rel" : "cdroms"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/snapshots",
    "rel" : "snapshots"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/tags",
    "rel" : "tags"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/permissions",
```

```
    "rel" : "permissions"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/statistics",
    "rel" : "statistics"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/reporteddevices",
    "rel" : "reporteddevices"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/watchdogs",
    "rel" : "watchdogs"
  }, {
    "href" : "/ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-
973a44b2562e/sessions",
    "rel" : "sessions"
  } ]
}
```

15.5. METHODS

15.5.1. Creating a Virtual Machine

Creating a new virtual machine requires the **name**, **template**, and **cluster** elements. Identify the **template** and **cluster** elements with the **id** attribute or **name** element. Identify the CPU profile ID with the **cpuprofiles** attribute.

Example 15.4. Creating a virtual machine with 512 MB that boots from CD-ROM

```
POST /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <name>vm2</name>
    <description>Virtual Machine 2</description>
    <type>desktop</type>
    <memory>536870912</memory>
    <cluster>
        <name>default</name>
    </cluster>
    <template>
        <name>Blank</name>
    </template>
    <os>
        <boot dev="cdrom"/>
    </os>
    <cdroms>
        <cdrom>
            <file id="example_windows_7_x64_dvd_u_677543.iso"/>
        </cdrom>
```

```
    </cdroms>
    <cpu_profile id="0000001a-001a-001a-001a-00000000035e"/>
</vm>
```

Example 15.5. Creating a virtual machine with 512 MB that boots from a virtual hard disk

```
POST /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <name>vm2</name>
    <description>Virtual Machine 2</description>
    <type>desktop</type>
    <memory>536870912</memory>
    <cluster>
        <name>default</name>
    </cluster>
    <template>
        <name>Blank</name>
    </template>
    <os>
      <boot dev="hd"/>
    </os>
    <cpu_profile id="0000001a-001a-001a-001a-00000000035e"/>
</vm>
```

NOTE

Memory in the previous example is converted to bytes using the following formula:

512MB * 1024^2 = 536870912 bytes

15.5.2. Updating a Virtual Machine

The **name**, **description**, **cluster**, **type**, **memory**, **cpu**, **os**, **high_availability**, **display**, **timezone**, **domain**, **stateless**, **placement_policy**, **memory_policy**, **usb**, **payloads**, **origin** and **custom_properties** elements are updatable post-creation.

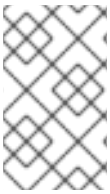
Example 15.6. Updating a virtual machine to contain 1 GB of memory

```
PUT /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <memory>1073741824</memory>
</vm>
```

NOTE

Memory in the previous example is converted to bytes using the following formula:

1024MB * 1024^2 = 1073741824 bytes



NOTE

Memory hot plug is supported in Red Hat Virtualization. If the virtual machine's operating system supports memory hot plug, you can use the example above to increase memory while the virtual machine is running.

Example 15.7. Hot plugging vCPUs

Add virtual CPUs to a running virtual machine without having to reboot it. In this example, the number of sockets is changed to 2.

```
PUT /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <cpu>
        <topology sockets="2" cores="1"/>
    </cpu>
</vm>
```

NOTE

CPU hot unplug is currently not supported in Red Hat Virtualization.

Example 15.8. Pinning a virtual machine to multiple hosts

A virtual machine that is pinned to multiple hosts cannot be live migrated, but in the event of a host failure, any virtual machine configured to be highly available is automatically restarted on one of the other hosts to which the virtual machine is pinned. Multi-host pinning can be used to restrict a virtual machine to hosts with, for example, the same hardware configuration.

```
PUT /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <high_availability>
        <enabled>true</enabled>
        <priority>1</priority>
    </high_availability>
    <placement_policy>
        <hosts>
            <host><name>Host1</name></host>
            <host><name>Host2</name></host>
        </hosts>
```

```
        <affinity>pinned</affinity>
    </placement_policy>
</vm>
```

15.5.3. Removing a Virtual Machine

Removal of a virtual machine requires a **DELETE** request.

Example 15.9. Removing a virtual machine

```
DELETE /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399
HTTP/1.1

HTTP/1.1 204 No Content
```

15.5.4. Removing a Virtual Machine but not the Virtual Disk

Detach the virtual disk prior to removing the virtual machine. This preserves the virtual disk. Removal of a virtual machine requires a **DELETE** request.

Example 15.10. Removing a virtual machine

```
DELETE /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <vm>
        <disks>
            <detach_only>true</detach_only>
        </disks>
    </vm>
</action>
```

15.6. SUB-COLLECTIONS

15.6.1. Disks Sub-Collection

15.6.1.1. Disks Sub-Collection

The **disks** sub-collection represents all virtual hard disk devices on a virtual machine. A **disk** representation contains the following elements:

Table 15.2. Elements for virtual disks

Element	Type	Description	Properties
link rel="statistics"	relationship	A link to the **statistics** sub-collection for a virtual machine's disk statistics.	
link rel="permissions"	relationship	A link to the **permissions** sub-collection.	
alias	string	The unique identifier for the disk. Use **alias** instead of **name**.	
image_id	string	A reference to the virtual machine image stored on the defined storage domain.	
storage_domains	complex	The storage domains associated with this disk. Each **storage_domain** element contains an **id** attribute with the associated storage domain's GUID. Update this element with **POST** to perform live migration of a disk from one data storage domain to another.	 [a]
size	integer	Size of the disk in bytes. Deprecated; replaced by **provisioned_size**.	
provisioned_size	integer	The provisioned size of the disk in bytes.	 
actual_size	integer	Actual size of the disk in bytes.	
status	One of **illegal**, **invalid**, **locked** or **ok**	The status of the disk device. These states are listed in **disk_states** under **capabilities**.	
interface	enumerated	The type of interface driver used to connect to the disk device. A list of enumerated values is available in **capabilities**.	

Element	Type	Description	Properties
format	enumerated	The underlying storage format. A list of enumerated values is available in **capabilities**. Copy On Write (COW) allows snapshots, with a small performance overhead. Raw does not allow snapshots, but offers improved performance.	
sparse	Boolean: true or false	**true** if the physical storage for the disk should not be preallocated.	
bootable	Boolean: true or false	**true** if this disk is to be marked as bootable.	
shareable	Boolean: true or false	**true** to share the disk with multiple virtual machines.	
wipe_after_delete	Boolean: true or false	**true** if the underlying physical storage for the disk should be zeroed when the disk is deleted. This increases security but is a more intensive operation and may prolong delete times.	
propagate_errors	Boolean: true or false	**true** if disk errors should not cause virtual machine to be paused and, instead, disk errors should be propagated to the guest OS.	
vm id=	GUID	The ID of the containing virtual machine.	
quota id=	GUID	Sets a quota for the disk.	
lun_storage	complex	A reference to a direct LUN mapping for storage usage. Requires a **logical_unit** element that contains iSCSI or FCP device details.	
active	Boolean	Defines if the disk is connected to the virtual machine.	
read_only	Boolean	Defines if the disk is read-only.	
link rel="disk_profile"	relationship	A link to the **disk_profile** sub-collection.	

Element	Type	Description	Properties

[a] This element is only required if the disk is being added to a virtual machine and not created from a virtual machine template.

Example 15.11. An XML representation of a disk device

```
<disk id="ed7feafe-9aaf-458c-809a-ed789cdbd5b4"
  href="/ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/disks/
  ed7feafe-9aaf-458c-809a-ed789cdbd5b4">
    <link rel="statistics"
      href="/ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/disks/
      ed7feafe-9aaf-458c-809a-ed789cdbd5b4/statistics"/>
    <link rel="permissions"
      href="/ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/disks/
      ed7feafe-9aaf-458c-809a-ed789cdbd5b4/permissions"/>
    <vm id="082c794b-771f-452f-83c9-b2b5a19c0399"
      href="/ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399"/>
    <alias>Classic_VM</alias>
    <image_id>cac69a29-ccff-49d4-8a26-e4cdacd83e34</image_id>
    <storage_domains>
        <storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"/>
    </storage_domains>
    <size>12884901888</size>
    <provisioned_size>12884901888</provisioned_size>
    <actual_size>1073741824</actual_size>
    <type>system</type>
    <status>
        <state>ok</state>
    </status>
    <interface>virtio</interface>
    <format>raw</format>
    <bootable>true</bootable>
    <shareable>true</shareable>
    <wipe_after_disk>true</wipe_after_disk>
    <propagate_errors>false</propagate_errors>
    <active>true</active>
    <read_only>false</read_only>
    <disk_profile id="23fb2e0d-3062-4819-8165-3be88f2f587e"
      href="/ovirt-engine/api/diskprofiles/23fb2e0d-3062-4819-8165-
3be88f2f587e"/>
    <lun_storage>
        <logical_unit id="lun1">
                ...
        </logical_unit>
    </lun_storage>
</disk>
```

Add a new virtual disk. When adding a new internal disk, the **provisioned_size** element is required. Use the **storage_domains** element to specify in which storage domain the disk will be created. Multiple disks for the same virtual machine can reside in different storage domains.

Example 15.12. Creating a new disk device on a virtual machine

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399/disks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <storage_domains>
        <storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"/>
    </storage_domains>
    <provisioned_size>8589934592</provisioned_size>
    <type>system</type>
    <interface>virtio</interface>
    <format>cow</format>
    <bootable>true</bootable>
</disk>
```

Add a new external (direct LUN) disk to a virtual machine. This method requires the **lun_storage** element and the **logical_unit** element, which contains iSCSI or FCP device details.

Example 15.13. Creating a new direct LUN disk device on a virtual machine

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399/disks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <interface>virtio</interface>
    <lun_storage>
        <type>iscsi</type>
        <logical_unit id="lun1">
            <address>iscsi.example.com</address>
            <port>3260</port>
            <target>iqn.2010.05.com.example:iscsi.targetX</target>
        </logical_unit>
    </lun_storage>
</disk>
```

The **alias**, **description**, **storage_domains**, **provisioned_size**, **interface**, **bootable**, **shareable**, **wipe_after_delete** and **propagate_errors** elements are updatable post-creation.

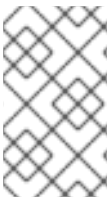
Users can resize virtual disks that are in use by one or more virtual machines, without pausing, hibernating or rebooting the virtual machine(s).

Example 15.14. Updating a virtual disk

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/disks/ed7feafe-9aaf-458c-809a-ed789cdbd5b4 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <bootable>false</bootable>
    <shareable>false</shareable>
</disk>
```

Example 15.15. Updating a virtual disk to 20GB

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/disks/ed7feafe-9aaf-458c-809a-ed789cdbd5b4 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <provisioned_size>21474836480</provisioned_size>
</disk>
```



NOTE

Disk size in the previous example is converted to bytes using the following formula:

20480MB * 1024^2 = 21474836480 bytes

Example 15.16. Renaming a virtual disk

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/disks/ed7feafe-9aaf-458c-809a-ed789cdbd5b4 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <alias>Classic_VM2</alias>
</disk>
```

Removal of a virtual disk requires a **DELETE** request.

Example 15.17. Removing a virtual disk

```
DELETE /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/disks/ed7feafe-9aaf-458c-809a-ed789cdbd5b4 HTTP/1.1

HTTP/1.1 204 No Content
```

15.6.1.2. Disk Cloning

Clone a disk from a template with the **clone** element. Set the **clone** element to **true** within the **disks** sub-collection when creating a virtual machine. This clones a disk from the base template and attaches it to the virtual machine.

Example 15.18. Cloning a disk from a template

The following example clones a disk from a template during the creation of a virtual machine.

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <name>cloned_vm</name>
    <template id="64d4aa08-58c6-4de2-abc4-89f19003b886"/>
    <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"/>
    <disks>
        <clone>true</clone>
        <disk id="4825ffda-a997-4e96-ae27-5503f1851d1b">
            <format>COW</format>
        </disk>
        <disk id="42aef10d-3dd5-4704-aa73-56a023c1464c">
            <format>COW</format>
        </disk>
    </disks>
</vm>
```

IMPORTANT

Search queries for virtual disks based upon disk name require the **alias** search parameter instead of **name**.

15.6.1.3. Disk Statistics Sub-Collection

Each virtual machine's disk exposes a **statistics** sub-collection for disk-specific statistics. Each **statistic** contains the following elements:

Table 15.3. Elements for virtual disk statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.

Element	Type	Description
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
disk id=	relationship	A relationship to the containing **disk** resource.

The following table lists the statistic types for virtual disks.

Table 15.4. Virtual disk statistic types

Name	Description
data.current.read	The data transfer rate in bytes per second when reading from the disk.
data.current.write	The data transfer rate in bytes per second when writing to the disk.

Example 15.19. An XML representation of a virtual machine's statistics sub-collection

```
<statistics>
    <statistic id="33b9212b-f9cb-3fd0-b364-248fb61e1272"
      href="/ovirt-engine/api/vms/3a42530e-3bc5-4094-829d-
489257894c2a/disks/
      f28ec14c-fc85-43e1-818d-96b49d50e27b/statistics/
      33b9212b-f9cb-3fd0-b364-248fb61e1272">
        <name>data.current.read</name>
        <description>Read data rate</description>
        <values type="DECIMAL">
            <value>
                <datum>0</datum>
            </value>
        </values>
        <type>GAUGE</type>
        <unit>BYTES_PER_SECOND</unit>
        <disk id="f28ec14c-fc85-43e1-818d-96b49d50e27b"
          href="/ovirt-engine/api/vms/3a42530e-3bc5-4094-829d-
489257894c2a/
          disks/f28ec14c-fc85-43e1-818d-96b49d50e27b"/>
    </statistic>
    ...
</statistics>
```

NOTE

This **statistics** sub-collection is read-only.

15.6.1.4. Floating Disk Attach and Detach Actions

Attach a disk from the main **rel="disks"** collection using a **POST** request on the virtual machine's **disks** sub-collection. Include the **id** of the disk to attach.

Example 15.20. Attach a floating disk

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399/disks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk id="d135f1c5-b5e1-4238-9381-b3277f5a3742">
</disk>
```

Detach a disk from a virtual machine's **disks** sub-collection using a **DELETE** request on the disk resource but ensure to include a **detach** Boolean element so the disk is not destroyed.

Example 15.21. Detach a disk from a virtual machine

```
DELETE /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399/disks/
  d135f1c5-b5e1-4238-9381-b3277f5a3742 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <detach>true</detach>
</action>
```

15.6.1.5. Disk Activate and Deactivate Actions

Each virtual machine's disk provides a set of **activate** and **deactivate** actions to add and remove disks from a virtual machine.

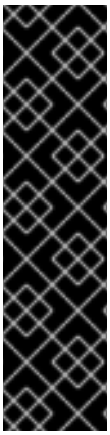
Example 15.22. Action to activate a virtual disk

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/disks/a42ada0e-1d69-410d-a392-a6980d873e5d/activate
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

Example 15.23. Action to deactivate a virtual disk

```
POST /ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/disks/a42ada0e-1d69-410d-a392-a6980d873e5d/deactivate
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

Use these actions to hotplug disks to virtual machines and activate newly attached floating disks.



IMPORTANT

The hotplugging feature only supports **VirtIO** disks and virtual machine operating systems that support hotplugging operations. Example operating systems include:

- Red Hat Enterprise Linux 6;
- Red Hat Enterprise Linux 5;
- Windows Server 2008; and,
- Windows Server 2003.

15.6.2. Network Interfaces Sub-Collection

15.6.2.1. Network Interfaces Sub-Collection

The **nics** sub-collection represents all network interface devices on a virtual machine. A **nic** representation contains the following elements:

Table 15.5. Elements for virtual machine network interfaces

Element	Type	Description	Properties
link rel="statistics"	relationship	A link to the **statistics** sub-collection for a virtual machine's network interface statistics.	
network id=	GUID	A reference to the network which the interface should be connected. A blank network id is allowed.	
interface	enumerated	The type of driver used for the nic. A list of enumerated values is available in **capabilities**.	
mac address=	string	The MAC address of the interface.	

Element	Type	Description	Properties
port_mirroring	complex	Defines whether the NIC receives mirrored traffic. Define a **networks** element with a series of **network id=** references.	
plugged	Boolean	Defines if the NIC is plugged in to the virtual machine.	
linked	Boolean	Defines if the NIC is linked to the virtual machine.	

Example 15.24. An XML representation of a network interface

```
<nic id="7a3cff5e-3cc4-47c2-8388-9adf16341f5e"
  ref="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-57d2af94f401/nics/
  7a3cff5e-3cc4-47c2-8388-9adf16341f5e">
    <link rel="statistics"
      href="/ovirt-engine/api/vms/082c794b-771f-452f-83c9-
b2b5a19c0399/nics/
      7a3cff5e-3cc4-47c2-8388-9adf16341f5e/statistics"/>
    <name>nic1</name>
    <interface>virtio</interface>
    <mac address="00:1a:4a:16:84:07"/>
    <network id="00000000-0000-0000-0000-000000000009"
      href="/ovirt-engine/api/networks/00000000-0000-0000-0000-
000000000009"/>
    <vm id="cdc0b102-fbfe-444a-b9cb-57d2af94f401"
      href="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401"/>
    <port_mirroring>
        <networks>
            <network id="56087282-d7a6-11e1-af44-001a4a400e0c"
              href="/ovirt-engine/api/networks/56087282-d7a6-11e1-af44-
001a4a400e0c"/>
        </networks>
    </port_mirroring>
</nic>
```

When adding a new network interface, the **name** and **network** elements are required. Identify the **network** element with the **id** attribute or **name** element.

Example 15.25. Creating a virtual machine NIC

```
POST /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-57d2af94f401/nics
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<nic>
```

```
    <name>nic1</name>
    <network id="00000000-0000-0000-0000-000000000009"/>
</nic>
```

An API user modifies a network interface with a **PUT** request.

Example 15.26. Updating a virtual machine NIC

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-57d2af94f401/nics/
7a3cff5e-3cc4-47c2-8388-9adf16341f5e HTTP/1.1
Accept: application/xml
Content-type: application/xml

<nic>
    <name>nic2</name>
    <network id="00000000-0000-0000-0000-000000000010"/>
    <type>e1000</type>
</nic>
```

An API user removes a network interface with a **DELETE** request.

Example 15.27. Deleting a virtual machine NIC

```
DELETE /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-57d2af94f401/nics/
7a3cff5e-3cc4-47c2-8388-9adf16341f5e HTTP/1.1

HTTP/1.1 204 No Content
```

IMPORTANT

The hotplugging feature only supports virtual machine operating systems with hotplugging operations. Example operating systems include:

- Red Hat Enterprise Linux 6;
- Red Hat Enterprise Linux 5;
- Windows Server 2008; and,
- Windows Server 2003.

15.6.2.2. Network Interface Statistics Sub-Collection

Each virtual machine's network interface exposes a **statistics** sub-collection for network interface statistics. Each **statistic** contains the following elements:

Table 15.6. Elements for a virtual machine's network interface statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
nic id=	relationship	A relationship to the containing **nic** resource.

The following table lists the statistic types for network interfaces on virtual machines.

Table 15.7. Virtual machine NIC statistic types

Name	Description
data.current.rx	The rate in bytes per second of data received.
data.current.tx	The rate in bytes per second of data transmitted.
errors.total.rx	Total errors from receiving data.
errors.total.tx	Total errors from transmitting data.

Example 15.28. An XML representation of a virtual machine's NIC statistics sub-collection

```
<statistics>
    <statistic id="ecd0559f-e88f-3330-94b4-1f091b0ffdf7"
      href="/ovirt-engine/api/vms/3a42530e-3bc5-4094-829d-
489257894c2a/nics/
      6cd08e76-57c0-41ba-a728-7eba46ae1e36/statistics/
      ecd0559f-e88f-3330-94b4-1f091b0ffdf7">
        <name>data.current.rx</name>
        <description>Receive data rate</description>
        <values type="DECIMAL">
            <value>
                <datum>0</datum>
            </value>
```

```
        </values>
        <type>GAUGE</type>
        <unit>BYTES_PER_SECOND</unit>
        <nic id="6cd08e76-57c0-41ba-a728-7eba46ae1e36"
          href="/ovirt-engine/api/vms/3a42530e-3bc5-4094-829d-
489257894c2a/
          nics/6cd08e76-57c0-41ba-a728-7eba46ae1e36"/>
    </statistic>
    ...
</statistics>
```

NOTE

This **statistics** sub-collection is read-only.

15.6.3. Virtual NUMA Nodes Sub-Collection

The **numanodes** sub-collection represents all virtual NUMA nodes on a virtual machine. A **vm_numa_node** representation contains the following elements:

Table 15.8. Elements for virtual NUMA nodes

Element	Type	Description	Properties
index	integer	The index number of the virtual NUMA node.	
memory	integer	The amount of memory allocated to the virtual NUMA node, in MB.	
cpu	complex	The CPU topology associated with this virtual NUMA node. Each **core** element contains an **index** attribute with the associated core's index number.	
vm id=	GUID	The ID of the containing virtual machine.	
numa_node_pins	complex	Pins the virtual NUMA node to a host NUMA node. Each **numa_node_pin** element contains a **pinned="true"** boolean and the host NUMA node's **index** number.	

Example 15.29. An XML representation of a virtual NUMA node

```
<vm_numa_node href="/ovirt-engine/api/vms/c7ecd2dc-dbd3-4419-956f-
1249651c0f2b/numanodes/3290b973-ed3e-4f0b-bbf5-9be10d229e50"
```

```
id="3290b973-ed3e-4f0b-bbf5-9be10d229e50">
        <index>0</index>
        <memory>1024</memory>
        <cpu>
            <cores>
                <core index="0"/>
            </cores>
        </cpu>
        <vm href="/ovirt-engine/api/vms/c7ecd2dc-dbd3-4419-956f-
1249651c0f2b" id="c7ecd2dc-dbd3-4419-956f-1249651c0f2b"/>
        <numa_node_pins>
            <numa_node_pin pinned="true" index="0">
                <host_numa_node id="417cdefb-8c47-4838-87f3-
dd0498fdf6c7"/>
            </numa_node_pin>
        </numa_node_pins>
</vm_numa_node>
```

When adding a new virtual NUMA node, the **index**, **memory**, and **cpu** elements are required.

Example 15.30. Adding a new virtual NUMA node to a virtual machine

```
POST /ovirt-engine/api/vms/c7ecd2dc-dbd3-4419-956f-
1249651c0f2b/numanodes HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm_numa_node>
  <index>0</index>
  <memory>1024</memory>
  <cpu>
    <cores>
      <core index="0"/>
    </cores>
  </cpu>
</vm_numa_nodes>
```

Update a virtual NUMA node with a **PUT** request. You can use a **PUT** request to pin a virtual NUMA node to a physical NUMA node on a host.

Example 15.31. Updating a virtual NUMA node

```
PUT /ovirt-engine/api/vms/c7ecd2dc-dbd3-4419-956f-
1249651c0f2b/numanodes/3290b973-ed3e-4f0b-bbf5-9be10d229e50 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm_numa_node>
  <numa_node_pins>
    <numa_node_pin pinned="true" index="0">
      <host_numa_node id="417cdefb-8c47-4838-87f3-dd0498fdf6c7"/>
```

```
    </numa_node_pin>
  </numa_node_pins>
</vm_numa_node>
```

Remove a virtual NUMA node with a **DELETE** request.

Example 15.32. Removing a virtual NUMA node

```
DELETE /ovirt-engine/api/vms/c7ecd2dc-dbd3-4419-956f-
1249651c0f2b/numanodes/3290b973-ed3e-4f0b-bbf5-9be10d229e50 HTTP/1.1

HTTP/1.1 204 No Content
```

15.6.4. CD-ROMs Sub-Collection

The **cdroms** sub-collection represents the CD-ROM device on a virtual machine. A **cdrom** representation contains the following elements:

Table 15.9. Elements for virtual machine CD-ROMs

Element	Type	Description	Properties
file id=	string/filename	A reference to an ISO image.	

Example 15.33. An XML representation of a CD-ROM device

```
<cdrom id="00000000-0000-0000-0000-000000000000"
  href="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/
  00000000-0000-0000-0000-000000000000">
    <file id="rhel-server-6.0-x86_64-dvd.iso"/>
    <vm id="cdc0b102-fbfe-444a-b9cb-57d2af94f401"
      href="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401"/>
</cdrom>
```

Send a **PUT** request with a **file id** element to add a new CD-ROM resource.

Example 15.34. Adding a new CD-ROM file

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/00000000-0000-0000-0000-000000000000 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cdrom>
    <file id="fedora-15-x86_64-dvd.iso"/>
</cdrom>
```

The API changes the CD-ROM using a **PUT** request:

Example 15.35. Changing a CD-ROM file

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/00000000-0000-0000-0000-000000000000 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cdrom>
    <file id="fedora-15-x86_64-dvd.iso"/>
</cdrom>
```

The API changes the CD-ROM for the current session only using a **PUT** request with an additional **current** URI argument:

Example 15.36. Changing a CD-ROM file during a current session

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/00000000-0000-0000-0000-000000000000;current=true
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<cdrom>
    <file id="fedora-15-x86_64-dvd.iso"/>
</cdrom>
```

To eject the CD-ROM temporarily, send a **PUT** request to the **cdroms** sub-collection of a virtual machine, adding the **current=true** matrix parameter:

Example 15.37. Ejecting a CD-ROM file during a current session

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/00000000-0000-0000-0000-000000000000;current=true
HTTP/1.1
Accept: application/xml
Content-type: application/xml
<cdrom>
  <file id=""/>
</cdrom>
```

NOTE

Rebooting the virtual machine will connect the CD-ROM again.

To eject the CD-ROM permanently, send a **PUT** request to the **cdroms** sub-collection of a virtual machine:

Example 15.38. Ejecting a CD-ROM file permanently

```
PUT /ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401/cdroms/00000000-0000-0000-0000-000000000000 HTTP/1.1
Accept: application/xml
Content-type: application/xml
<cdrom>
  <file id=""/>
</cdrom>
```

NOTE

Virtual machines only contain a single CD-ROM device.

15.6.5. Snapshots Sub-Collection

15.6.5.1. Snapshots Sub-Collection

A virtual machine saves and restores disk state as a number of snapshots. These are represented and managed through a **rel="snapshot"** sub-collection that behaves similar to other collections.

Each virtual machine snapshot is represented with an individual **snapshot** element that contains the following sub-elements:

Table 15.10. Elements for virtual machine snapshots

Element	Type	Description	Properties
vm id=	GUID	The ID and URI of the virtual machine to which this snapshot pertains.	
link rel="restore"	relationship	A link to restore the snapshot of the virtual machine.	
link rel="prev"	relationship	A link to the previous snapshot of this virtual machine.	
type	string	The type of the snapshot. For example, **active** or **regular**.	
date	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The date and time at which the snapshot was created.	
snapshot_status	string	The current status of the snapshot.	

Element	Type	Description	Properties
persist_memorystate	Boolean	Defines whether the snapshot also includes the state of the memory of the virtual machine at the time the snapshot was taken.	

NOTE

It is not possible to modify snapshot elements using **PUT**.

Example 15.39. An XML representation of a virtual machine snapshot

```
<snapshot id="00000000-0000-0000-0000-000000000000"
  href="/ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000/snapshots/
  00000000-0000-0000-0000-000000000000">
    <actions>
      <link rel="restore"
      href="/ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000/snapshots/
      00000000-0000-0000-0000-000000000000/restore"/>
    <link rel="prev"
      href="/ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000/snapshots/
    </actions>
    <vm id="00000000-0000-0000-0000-000000000000"
      href="/ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000"/>
    <description>Virtual Machine 1 - Snapshot A</description>
    <type>active</type>
    <date>2010-08-16T14:24:29</date>
    <snapshot_status>ok</snapshot_status>
    <persist_memorystate>false</persist_memorystate>
</snapshot>
```

Use a **GET** request for a virtual machine snapshot with the **All-Content: true** header to include additional OVF data with the representation of the snapshot.

The **Accept** header defaults to **application/xml** if left blank, and the data is represented with HTML entities so as not to interfere with the XML tags. Specifying the **Accept: application/json** header will return the data in standard XML tagging. This example representation has been formatted from its standard block format to improve legibility.

Example 15.40. XML representation of additional ovf data for a snapshot

```
GET /ovirt-engine/api/vms/42ec2621-7ad6-4ca2-bd68-973a44b2562e/snapshots
HTTP/1.1
All-Content: true

<?xml version='1.0' encoding='UTF-8'?>
```

```
<ovf:Envelope xmlns:ovf=\"http://schemas.dmtf.org/ovf/envelope/1/\"
  xmlns:rasd=\"http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/CIM_ResourceAllocationSettingData\"
  xmlns:vssd=\"http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/CIM_VirtualSystemSettingData\"
  xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
ovf:version=\"3.5.0.0\">
  <References>
  <File ovf:href=\"ad353554-f668-46cf-aa3c-e57383de2c92/40456d92-3687-
4a85-bab3-87b4cc7af459\"
    ovf:id=\"40456d92-3687-4a85-bab3-87b4cc7af459\"
ovf:size=\"10737418240\"
    ovf:description=\"Active VM\"/>
  <Nic ovf:id=\"be14bfc8-3dbd-4ac1-ba02-c6dfa7fc707c\"/>
  </References>
  <Section xsi:type=\"ovf:NetworkSection_Type\">
    <Info>List of networks</Info><Network ovf:name=\"Network 1\"/>
  </Section>
  <Section
    xsi:type=\"ovf:DiskSection_Type\">
    <Info>List of Virtual Disks</Info>
    <Disk ovf:diskId=\"40456d92-3687-4a85-bab3-87b4cc7af459\"
    ovf:size=\"10\" ovf:actual_size=\"0\"
    ovf:vm_snapshot_id=\"a209216d-2909-4802-8886-02aad55dccc8\"
    ovf:parentRef=\"\"
    ovf:fileRef=\"ad353554-f668-46cf-aa3c-e57383de2c92/40456d92-3687-
4a85-bab3-87b4cc7af459\"
    ovf:format=\"http://www.vmware.com/specifications/vmdk.html#sparse\"
    ovf:volume-format=\"RAW\"
    ovf:volume-type=\"Preallocated\"
    ovf:disk-interface=\"VirtIO\"
    ovf:boot=\"true\"
    ovf:disk-alias=\"VM_01_Disk1\"
    ovf:wipe-after-delete=\"false\"/>
  </Section>
  <Content
    ovf:id=\"out\"
    xsi:type=\"ovf:VirtualSystem_Type\">
    <CreationDate>2015/02/09 13:53:53</CreationDate>
    <ExportDate>2015/02/10 00:39:24</ExportDate>
    <DeleteProtected>false</DeleteProtected>
    <SsoMethod>guest_agent</SsoMethod>
    <IsSmartcardEnabled>false</IsSmartcardEnabled>
    <TimeZone>Etc/GMT</TimeZone>
<default_boot_sequence>0</default_boot_sequence>
    <Generation>1</Generation>
    <VmType>1</VmType>
    <MinAllocatedMem>1024</MinAllocatedMem>
    <IsStateless>false</IsStateless>
    <IsRunAndPause>false</IsRunAndPause>
    <AutoStartup>false</AutoStartup>
    <Priority>1</Priority>
    <CreatedByUserId>fdfc627c-d875-11e0-90f0-
83df133b58cc</CreatedByUserId>
    <IsBootMenuEnabled>false</IsBootMenuEnabled>
    <IsSpiceFileTransferEnabled>true</IsSpiceFileTransferEnabled>
```

```
    <IsSpiceCopyPasteEnabled>true</IsSpiceCopyPasteEnabled>
    <Name>VM_01</Name>
    <TemplateId>00000000-0000-0000-0000-000000000000</TemplateId>
    <TemplateName>Blank</TemplateName>
    <IsInitilized>true</IsInitilized>
    <Origin>3</Origin>
    <DefaultDisplayType>1</DefaultDisplayType>
    <TrustedService>false</TrustedService>
    <OriginalTemplateId>00000000-0000-0000-0000-
000000000000</OriginalTemplateId>
    <OriginalTemplateName>Blank</OriginalTemplateName>
    <UseLatestVersion>false</UseLatestVersion>
    <Section ovf:id=\"42ec2621-7ad6-4ca2-bd68-973a44b2562e\"
ovf:required=\"false\" xsi:type=\"ovf:OperatingSystemSection_Type\">
      <Info>Guest Operating System</Info>
      <Description>other</Description>
    </Section>
    <Section xsi:type=\"ovf:VirtualHardwareSection_Type\">
      <Info>1 CPU, 1024 Memeory</Info>
      <System>
        <vssd:VirtualSystemType>ENGINE 3.5.0.0</vssd:VirtualSystemType>
      </System>
      <Item>
        <rasd:Caption>1 virtual cpu</rasd:Caption>
        <rasd:Description>Number of virtual CPU</rasd:Description>
        <rasd:InstanceId>1</rasd:InstanceId>
        <rasd:ResourceType>3</rasd:ResourceType>
        <rasd:num_of_sockets>1</rasd:num_of_sockets>
        <rasd:cpu_per_socket>1</rasd:cpu_per_socket>
      </Item>
      <Item>
        <rasd:Caption>1024 MB of memory</rasd:Caption>
        <rasd:Description>Memory Size</rasd:Description>
        <rasd:InstanceId>2</rasd:InstanceId>
        <rasd:ResourceType>4</rasd:ResourceType>
        <rasd:AllocationUnits>MegaBytes</rasd:AllocationUnits>
        <rasd:VirtualQuantity>1024</rasd:VirtualQuantity>
      </Item>
      <Item>
        <rasd:Caption>VM_01_Disk1</rasd:Caption>
        <rasd:InstanceId>40456d92-3687-4a85-bab3-
87b4cc7af459</rasd:InstanceId>
        <rasd:ResourceType>17</rasd:ResourceType>
        <rasd:HostResource>ad353554-f668-46cf-aa3c-
e57383de2c92/40456d92-3687-4a85-bab3-87b4cc7af459</rasd:HostResource>
        <rasd:Parent>00000000-0000-0000-0000-000000000000</rasd:Parent>
        <rasd:Template>00000000-0000-0000-0000-
000000000000</rasd:Template>
        <rasd:ApplicationList></rasd:ApplicationList>
        <rasd:StoragePoolId>00000002-0002-0002-0002-
000000000255</rasd:StoragePoolId>
        <rasd:CreationDate>2015/02/09 13:54:41</rasd:CreationDate>
        <rasd:LastModified>1970/01/01 00:00:00</rasd:LastModified>
        <rasd:last_modified_date>2015/02/10
00:39:22</rasd:last_modified_date>
        <Type>disk</Type>
```

```
        <Device>disk</Device>
        <rasd:Address>{slot=0x06, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>1</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>virtio-disk0</Alias>
      </Item>
      <Item>
        <rasd:Caption>Ethernet adapter on ovirtmgmt</rasd:Caption>
        <rasd:InstanceId>be14bfc8-3dbd-4ac1-ba02-
c6dfa7fc707c</rasd:InstanceId>
        <rasd:ResourceType>10</rasd:ResourceType>
        <rasd:OtherResourceType>ovirtmgmt</rasd:OtherResourceType>
        <rasd:ResourceSubType>3</rasd:ResourceSubType>
        <rasd:Connection>ovirtmgmt</rasd:Connection>
        <rasd:Linked>true</rasd:Linked>
        <rasd:Name>nic1</rasd:Name>
        <rasd:MACAddress>00:1a:4a:87:cb:00</rasd:MACAddress>
        <rasd:speed>1000</rasd:speed>
        <Type>interface</Type>
        <Device>bridge</Device>
        <rasd:Address>{slot=0x03, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>net0</Alias>
      </Item>
      <Item>
        <rasd:Caption>USB Controller</rasd:Caption>
        <rasd:InstanceId>3</rasd:InstanceId>
        <rasd:ResourceType>23</rasd:ResourceType>
        <rasd:UsbPolicy>DISABLED</rasd:UsbPolicy>
      </Item>
      <Item>
        <rasd:Caption>Graphical Controller</rasd:Caption>
        <rasd:InstanceId>17bbf0db-7cf0-4529-9b53-
dee6dee41cfd</rasd:InstanceId>
        <rasd:ResourceType>20</rasd:ResourceType>
        <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
        <rasd:SinglePciQxl>false</rasd:SinglePciQxl>
        <Type>video</Type>
        <Device>qxl</Device>
        <rasd:Address>{slot=0x02, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>true</IsReadOnly>
        <Alias>video0</Alias>
        <SpecParams>
          <vram>32768</vram>
          <heads>1</heads>
        </SpecParams>
      </Item>
      <Item>
```

```
        <rasd:Caption>CDROM</rasd:Caption>
        <rasd:InstanceId>7ce1bd14-d98a-43ba-beee-
520bdfd9c698</rasd:InstanceId>
        <rasd:ResourceType>15</rasd:ResourceType>
        <Type>disk</Type>
        <Device>cdrom</Device>
        <rasd:Address>{bus=1, controller=0, type=drive, target=0,
unit=0}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>true</IsReadOnly>
        <Alias>ide0-1-0</Alias></Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>8758c42f-7523-461b-82bb-
41d91e46fd36</rasd:InstanceId>
        <Type>controller</Type>
        <Device>usb</Device>
        <rasd:Address>{slot=0x01, bus=0x00, domain=0x0000, type=pci,
function=0x2}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>usb0</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>58f1a596-553e-4e95-9331-
64b5d8cebe2e</rasd:InstanceId>
        <Type>controller</Type>
        <Device>ide</Device>
        <rasd:Address>{slot=0x01, bus=0x00, domain=0x0000, type=pci,
function=0x1}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>ide0</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>2f4f8aa5-25eb-4a31-b841-
50dc48fce4a7</rasd:InstanceId>
        <Type>channel</Type>
        <Device>unix</Device>
        <rasd:Address>{bus=0, controller=0, type=virtio-serial, port=1}
</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>channel0</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>edaac3ed-2ab6-48b1-ae77-
cc98f8b45bd8</rasd:InstanceId>
        <Type>channel</Type>
```

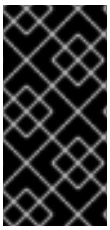
```
        <Device>unix</Device>
        <rasd:Address>{bus=0, controller=0, type=virtio-serial, port=2}
</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>channel1</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>8dfed248-5164-41d3-8b6e-
46aef9798d84</rasd:InstanceId>
        <Type>channel</Type>
        <Device>spicevmc</Device>
        <rasd:Address>{bus=0, controller=0, type=virtio-serial, port=3}
</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>channel2</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>d184185e-ee19-442a-88f5-
6a48f14164e1</rasd:InstanceId>
        <Type>controller</Type>
        <Device>virtio-scsi</Device>
        <rasd:Address>{slot=0x04, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>scsi0</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>374d219e-e2ff-4755-a544-
d537c87e82df</rasd:InstanceId>
        <Type>controller</Type>
        <Device>virtio-serial</Device>
        <rasd:Address>{slot=0x05, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>0</BootOrder>
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>false</IsReadOnly>
        <Alias>virtio-serial0</Alias>
      </Item>
      <Item>
        <rasd:ResourceType>0</rasd:ResourceType>
        <rasd:InstanceId>cf3d7121-9db0-4fd1-bd12-
50ce4e1ce379</rasd:InstanceId>
        <Type>balloon</Type>
        <Device>memballoon</Device>
        <rasd:Address>{slot=0x07, bus=0x00, domain=0x0000, type=pci,
function=0x0}</rasd:Address>
        <BootOrder>0</BootOrder>
```

```
        <IsPlugged>true</IsPlugged>
        <IsReadOnly>true</IsReadOnly>
        <Alias>balloon0</Alias>
        <SpecParams>
          <model>virtio</model>
        </SpecParams>
      </Item>
    </Section>
  </Content>
</ovf:Envelope>
```

You can create a snapshot of a virtual machine that is running (a live snapshot) or shut down by using the **POST** method:

Example 15.41. Creating a Virtual Machine Snapshot

```
POST /ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000/snapshots/ HTTP/1.1
Accept: application/xml
Content-type: application/xml

<snapshot>
<description>Snapshot description</description>
</snapshot>
```



IMPORTANT

Before taking a live snapshot of a virtual machine that uses OpenStack Volume (Cinder) disks, you must freeze and thaw the guest filesystem manually. See Section 15.7.14, “Freeze Virtual Machine Filesystems Action” and Section 15.7.15, “Thaw Virtual Machine Filesystems Action” for more information.

You can restore a virtual machine snapshot using the **`rel="restore"`** action link in the snapshot representation:

Example 15.42. Restoring a Virtual Machine Snapshot

```
POST /ovirt-engine/api/vms/00000000-0000-0000-0000-
000000000000/snapshots/00000000-0000-0000-0000-000000000000/restore
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.6.5.2. Clone a Virtual Machine from a Snapshot

API provides a function to create virtual machines from a snapshot of a previous machine. API users create a new virtual machine while retaining the original virtual machine with all snapshots intact.

Creation of a virtual machines from a snapshot requires an additional **snapshots** element to a standard representation of a virtual machine, which a user sends in a **POST** request to the **vms** collection.

The **snapshots** element contains a **snapshot id=** element to define the specific snapshot to use as a basis for the virtual machine.

Example 15.43. Clone Virtual Machine from Snapshot

```
POST /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
  ...
  <snapshots>
    <snapshot id="3f68ee63-0016-4f8c-9b8a-11d9f28f7c9e"/>
  </snapshots>
  ...
</vm>
```

15.6.6. Statistics Sub-Collection

Each virtual machine resource exposes a **statistics** sub-collection for virtual machine-specific statistics. Each **statistic** contains the following elements:

Table 15.11. Elements for virtual machine statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
vm id=	relationship	A relationship to the containing **vm** resource.

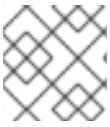
The following table lists the statistic types for virtual machines.

Table 15.12. Virtual machine statistic types

Name	Description
memory.installed	Total memory in bytes allocated for the virtual machine's use.
memory.used	Current memory in bytes used by the virtual machine.
cpu.current.guest	Percentage of CPU used by the guest.
cpu.current.hypervisor	Percentage of CPU overhead on the hypervisor.
cpu.current.total	Total percentage of the current CPU in use.

Example 15.44. An XML representation of a virtual machine's statistics sub-collection

```
<statistics>
    <statistic id="ef802239-b74a-329f-9955-be8fea6b50a4"
      href="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-57d2af94f401/
      statistics/ef802239-b74a-329f-9955-be8fea6b50a4">
        <name>memory.installed</name>
        <description>Total memory configured</description>
        <unit>BYTES</unit>
        <type>GUAGE</type>
        <values type="DECIMAL">
            <value>
                <datum>1073741824<datum>
            </value>
        </values>
        <vm id="cdc0b102-fbfe-444a-b9cb-57d2af94f401"
          href="/ovirt-engine/api/vms/cdc0b102-fbfe-444a-b9cb-
57d2af94f401"/>
    </statistic>
    ...
</statistics>
```



NOTE

A virtual machine's **statistics** sub-collection is read-only.

15.6.7. Displaying Virtual Machine Session Information

Submit a **GET** request for a virtual machine and use the **session** sub-collection to view the session information for the user that initiated the SPICE console session and the user logged in to the virtual machine.

The **session** information of a virtual machine is listed as a sub-collection:

Example 15.45. Displaying the session information of a virtual machine

```
GET /ovirt-engine/api/roles/a1a701f1-aa06-4f02-af17-
```

```
158be31489b3/sessions HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<sessions>
  <session id="37a6259c-c0c1-dae2-99a7-866489dff0bd"
    href= "/ovirt-engine/api/vms/a1a701f1-aa06-4f02-af17-
158be31489b3/sessions/37a6259c-c0c1-dae2-99a7-866489dff0bd">
  <vm href= "/ovirt-engine/api/vms/a1a701f1-aa06-4f02-af17-158be31489b3"
id="a1a701f1-aa06-4f02-af17-158be31489b3"/>
  <ip address="192.0.2.0"/>
  <user href= "/ovirt-engine/api/users/fdfc627c-d875-11e0-90f0-
83df133b58cc" id="fdfc627c-d875-11e0-90f0-83df133b58cc">
    <domain href= "/ovirt-engine/api/domains/696e7465-726e-616c-696e-
7465726e616c" id="696e7465-726e-616c-696e-7465726e616c">
      <name>internal</name>
    </domain>
    <user_name>admin</user_name>
    </user>
    <console_user>true</console_user>
  </session>
  <session id="37a6259c-c0c1-dae2-99a7-866489dff0bd"
    href= "/ovirt-engine/api/vms/a1a701f1-aa06-4f02-af17-
158be31489b3/sessions/37a6259c-c0c1-dae2-99a7-866489dff0bd" >
    <vm href= "/ovirt-engine/api/vms/a1a701f1-aa06-4f02-af17-
158be31489b3" id="a1a701f1-aa06-4f02-af17-158be31489b3"/>
    <user>
      <user_name>root</user_name>
    </user>
  </session>
</sessions>
```

15.7. ACTIONS

15.7.1. Start Virtual Machine Action

The start action launches a stopped, shutdown, or suspended virtual machine.

Example 15.46. Action to start a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/start
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

The start action allows a **vm** element to be provided as a parameter. If a **vm** element is provided, the virtual machine uses the values from the provided element and overrides system settings at start time.

Using the start action with the **vm** element in REST API is equivalent to using the **Run Once** window in the Administration or User Portal. These settings persist until a user stops the virtual machine. Examples of these elements include **os**, **domain**, **placement_policy**, **cdroms**, **stateless** and **display type**.

Example 15.47. Action to start a virtual machine with overridden parameters

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/start
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <pause>true</pause>
    <vm>
        <stateless>true</stateless>
        <display>
            <type>spice</type>
        </display>
        <os>
            <boot dev="cdrom"/>
        </os>
        <cdroms>
            <cdrom>
                <file id="windows-xp.iso"/>
            </cdrom>
        </cdroms>
        <floppies>
            <floppy>
                <file id="virtio-win_x86.vfd"/>
            </floppy>
       </floppies>
        <domain>
            <name>domain.example.com</name>
            <user>
                <user_name>domain_user</user_name>
                <password>domain_password</password>
            </user>
        </domain>
        <placement_policy>
            <host id="02447ac6-bcba-448d-ba2b-f0f453544ed2"/>
        </placement_policy>
    </vm>
</action>
```

NOTE

- The **domain** element is used for Windows systems only for overriding parameters on boot with the **start** action. The **domain** element determines the domain that the Windows virtual machine joins. If the domain does not exist in the **domains** collection, this element requires additional **user** authentication details, including a **user_name** and **password**. If the domain exists in the **domains** collection, the action requires no additional **user** authentication details.
- The CD image and floppy disk file must be available in the ISO domain already. If not, use the ISO uploader tool to upload the files. See The ISO Uploader Tool for more information.

15.7.2. Start Virtual Machine with Cloud-Init Action

Cloud-Init is a tool for automating the initial setup of virtual machines. You can use the tool to configure the host name, network interfaces, the DNS service, authorized keys, and set user names and passwords. You can also use the **custom_script** tag to specify a custom script to run on the virtual machine when it boots.

NOTE

The **cloud-init** element can only be used to start virtual machines with the cloud-init package installed. When the **cloud-init** element is used, any element within the **initialization** element but outside the **cloud-init** element will be ignored.

Example 15.48. Action to start a virtual machine using Cloud-Init

This example shows you how to start a virtual machine using the Cloud-Init tool to set the host name, change the root password, set a static IP for the **eth0** interface, configure DNS, and add an SSH key for the **root** user.

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/start
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <vm>
       <initialization>
        <cloud_init>
          <host>
           <address>MyHost.MyDomain.com</address>
          </host>
          <users>
           <user>
            <user_name>root</user_name>
            <password>p@55w0rd!</password>
           </user>
          </users>
          <network_configuration>
           <nics>
            <nic>
              <name>eth0</name>
```

```
              <boot_protocol>static</boot_protocol>
              <network>
                <ip address="192.168.122.31" netmask="255.255.255.0"
gateway="192.168.122.1"/>
              </network>
              <on_boot>true</on_boot>
             </nic>
            </nics>
            <dns>
             <servers>
               <host>
                 <address>192.168.122.1</address>
               </host>
             </servers>
             <search_domains>
               <host>
                 <address>MyDomain.com</address>
               </host>
             </search_domains>
           </dns>
         </network_configuration>
         <authorized_keys>
          <authorized_key>
            <user>
              <user_name>root</user_name>
            </user>
            <key>ssh-rsa AAAAB3Nza[...]75zkdD root@MyDomain.com</key>
          </authorized_key>
         </authorized_keys>
        </cloud_init>
        <custom_script><![CDATA[your script]]></custom_script>
       </initialization>
  </vm>
</action>
```

15.7.3. Stop Virtual Machine Action

The stop action forces a virtual machine to power-off.

Example 15.49. Action to stop a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/stop
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.7.4. Shutdown Virtual Machine Action

The shutdown action sends a shutdown request to a virtual machine.

Example 15.50. Action to send a shutdown request to a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/shutdown
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.7.5. Suspend Virtual Machine Action

The suspend action saves the virtual machine state to disk and stops it. Start a suspended virtual machine and restore the virtual machine state with the start action.

Example 15.51. Action to save virtual machine state and suspend the machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/suspend
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.7.6. Reboot Virtual Machine Action

The reboot action sends a reboot request to a virtual machine.

Example 15.52. Action to send a reboot request to a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/reboot
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.7.7. Enable user logon to access a virtual machine from an external console

The logon action enables users to access a virtual machine from consoles outside of the Red Hat Virtualization environment.

This action requires the ovirt-guest-agent-gdm-plugin and the ovirt-guest-agent-pam-module packages to be installed and the **`ovirt-guest-agent`** service to be running on the virtual machine.

Users require the appropriate user permissions for the virtual machine in order to access the virtual machine from an external console.

Example 15.53. Logging onto a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/logon
HTTP/1.1
Content-Type: application/json
Content-Length: 2

{}
```

15.7.8. Detach Virtual Machine from Pool Action

The detach action detaches a virtual machine from a pool.

Example 15.54. Action to detach a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/detach
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

15.7.9. Migrate Virtual Machine Action

The migrate action migrates a virtual machine to another physical host. The destination **`host`** element is an optional element as Red Hat Virtualization Manager automatically selects a default host for migration. If an API user requires a specific **`host`**, the user can specify the host with either an **`id`** or **`name`** parameter.

Example 15.55. Action to migrate a virtual machine to another host

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/migrate
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <host id="2ab5e1da-b726-4274-bbf7-0a42b16a0fc3"/>
</action>
```

15.7.10. Cancel Virtual Machine Migration Action

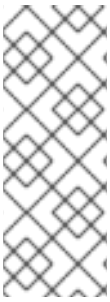
The cancel migration action stops any migration of a virtual machine to another physical host.

Example 15.56. Action to cancel migration of a virtual machine to another host

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/cancelmigration HTTP/1.1
Accept: application/xml
```

```
Content-type: application/xml

<action/>
```

15.7.11. Export Virtual Machine Action



NOTE

The export storage domain is deprecated. Storage data domains can be unattached from a data center and imported to another data center in the same environment, or in a different environment. Virtual machines, floating virtual disk images, and templates can then be uploaded from the imported storage domain to the attached data center. See the Importing Existing Storage Domains section in the *Red Hat Virtualization Administration Guide* for information on importing storage domains.

The export action exports a virtual machine to an **`export`** storage domain. A destination storage domain must be specified with a **`storage_domain`** reference.

The export action reports a failed action if a virtual machine of the same name exists in the destination domain. Set the **`exclusive`** parameter to **`true`** to change this behavior and overwrite any existing virtual machine.

If snapshots of the virtual machine are not included with the exported virtual machine, set the **`discard_snapshots`** parameter to **`true`**.

Example 15.57. Action to export a virtual machine to an export storage domain

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/export
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain>
        <name>export1</name>
    </storage_domain>
    <exclusive>true</exclusive>
    <discard_snapshots>true</discard_snapshots>
</action>
```

15.7.12. Virtual Machine Ticket Action

The ticket action generates a time-sensitive authentication token for accessing a virtual machine's display. The client-provided **`action`** optionally includes a **`ticket`** representation containing a **`value`** (if the token string needs to take on a particular form) and/or an **`expiry`** time in minutes. In any case, the response specifies the actual ticket value and expiry used.

Example 15.58. Action to generate authentication token for a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/ticket
```

```
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <ticket>
        <expiry>120</expiry>
    </ticket>
</action>

200 OK
Content-Type: application/xml

<action id="94e07552-14ba-4c27-8ce6-2cc75190d3ef"
  href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/ticket/
  94e07552-14ba-4c27-8ce6-2cc75190d3ef">
    <status>
        <state>complete</state>
    </status>
    <ticket>
        <value>5c7CSzK8Sw41</value>
        <expiry>120</expiry>
    </ticket>
    <link rel="parent"
      href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720"/>
    <link rel="replay"
      href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/ticket"/>
</action>
```

15.7.13. Force Remove Virtual Machine Action

An API user forces the removal of a faulty virtual machine with the **`force`** action. This action requires a **`DELETE`** method. The request body contains an **`action`** representation with the **`force`** parameter set to **`true`**. The request also requires an additional **`Content-type: application/xml`** header to process the XML representation in the body.

Example 15.59. Force remove action on a virtual machine

```
DELETE /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <force>true</force>
</action>
```

15.7.14. Freeze Virtual Machine Filesystems Action

The **freezefilesystems** action freezes a virtual machine's filesystems using the QEMU guest agent when taking a live snapshot of a running virtual machine. Normally, this is done automatically by the Manager, but this must be executed manually with the REST API for virtual machines using OpenStack Volume (Cinder) disks.

Freezing the filesystems on the guest operating system ensures a consistent snapshot. Once the snapshot is finished, the guest filesystems must then be thawed. On virtual machines not using a OpenStack Volume disk, the freezing and thawing actions can also be invoked manually using the REST API, which can be useful in the case of a failure during the snapshot process.

Example 15.60. Action to freeze a virtual machine's filesystems

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/freezefilesystems HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

For more information on snapshots, see Section 15.6.5.1, “Snapshots Sub-Collection” or the Snapshots section in the *Red Hat Virtualization Virtual Machine Management Guide*.

15.7.15. Thaw Virtual Machine Filesystems Action

The **thawfilesystems** action thaws a virtual machine's filesystems using the QEMU guest agent when taking a live snapshot of a running virtual machine. Normally, this is done automatically by the Manager, but this must be executed manually with the REST API for virtual machines using OpenStack Volume (Cinder) disks.

Freezing the filesystems on the guest operating system ensures a consistent snapshot. Once the snapshot is finished, the guest filesystems must then be thawed. On virtual machines not using a OpenStack Volume disk, the freezing and thawing actions can also be invoked manually using the REST API, which can be useful in the case of a failure during the snapshot process. For example, if the virtual machine became unresponsive during thaw, you can execute the thaw operation again manually; otherwise the virtual machine may remain unresponsive.

Example 15.61. Action to thaw a virtual machine's filesystems

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/thawfilesystems HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

For more information on snapshots, see Section 15.6.5.1, “Snapshots Sub-Collection” or the Snapshots section in the *Red Hat Virtualization Virtual Machine Management Guide*.

CHAPTER 16. FLOATING DISKS

16.1. FLOATING DISK ELEMENTS

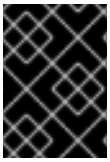
The **disks** collection provides information about all disks in a Red Hat Virtualization environment. A user attaches and detaches disks from any virtual machine and floats them between virtual machines. An API user accesses this information through the **rel="disks"** link obtained from the entry point URI.

The following table shows specific elements contained in a **disks** resource representation.

Table 16.1. Elements for floating disks

Element	Type	Description	Properties
link rel="statistics"	relationship	A link to the **statistics** sub-collection for a virtual machine's disk statistics.	
image_id	GUID	A reference to the virtual machine image stored on the defined storage domain.	
storage_domains	Complex	The storage domains associated with this disk. Each **storage_domain** element contains an **id** attribute with the associated storage domain's GUID. Update this element with **POST** to perform live migration of a disk from one data storage domain to another.	
size	integer	Size of the disk in bytes.	
provisioned_size	integer	The provisioned size of the disk in bytes.	 
actual_size	integer	Actual size of the disk in bytes.	
status	One of **illegal**, **invalid**, **locked** or **ok**	The status of the disk device. These states are listed in **disk_states** under **capabilities**.	
interface	enumerated	The type of interface driver used to connect to the disk device. A list of enumerated values is available in **capabilities**.	

Element	Type	Description	Properties
format	enumerated	The underlying storage format. A list of enumerated values is available in **capabilities**. Copy On Write (COW) allows snapshots, with a small performance overhead. Raw does not allow snapshots, but offers improved performance.	
sparse	Boolean: true or false	**true** if the physical storage for the disk should not be preallocated.	
bootable	Boolean: true or false	**true** if this disk is to be marked as bootable.	
shareable	Boolean: true or false	**true** to share the disk with multiple virtual machines.	
wipe_after_delete	Boolean: true or false	**true** if the underlying physical storage for the disk should be zeroed when the disk is deleted. This increases security but is a more intensive operation and may prolong delete times.	
propagate_errors	Boolean: true or false	**true** if disk errors should not cause virtual machine to be paused and, instead, disk errors should be propagated to the guest OS.	
quota id=	GUID	Sets a quota for the disk.	
lunStorage	complex	A reference to a direct LUN mapping for storage usage. Requires a **storage** element that contains iSCSI or FCP device details.	
active	Boolean	Defines if the disk is connected to the virtual machine.	



IMPORTANT

Search queries for disks based upon name require the **alias** search parameter instead of **name**.

16.2. XML REPRESENTATION OF A FLOATING DISK

Example 16.1. An XML representation of a disk device

```
<disk id="ed7feafe-9aaf-458c-809a-ed789cdbd5b4"
  href="/ovirt-engine/api/disks/ed7feafe-9aaf-458c-809a-ed789cdbd5b4">
    <link rel="statistics"
      href="/ovirt-engine/api/disks/ed7feafe-9aaf-458c-809a-
ed789cdbd5b4/statistics"/>
    <storage_domains>
        <storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"/>
    </storage_domains>
    <size>10737418240</size>
    <type>system</type>
    <status>
        <state>ok</state>
    </status>
    <interface>virtio</interface>
    <format>raw</format>
    <bootable>true</bootable>
    <shareable>true</shareable>
    <lunStorage>
        <storage>
            <logical_unit id="lun1">
                ...
            </logical_unit>
        <storage>
    </lunStorage>
</disk>
```

16.3. METHODS

16.3.1. Creating a Floating Disk

When creating a new floating disk, the API requires the **`size`** and **`storage_domains`** elements.

Example 16.2. Creating a new a floating disk device

```
POST /ovirt-engine/api/disks HTTP/1.1
Accept: application/xml
Content-type: application/xml

<disk>
    <storage_domains>
        <storage_domain id="fabe0451-701f-4235-8f7e-e20e458819ed"/>
    </storage_domains>
    <size>8589934592</size>
    <type>system</type>
    <interface>virtio</interface>
    <format>cow</format>
</disk>
```

16.4. SUB-COLLECTIONS

16.4.1. Statistics Sub-Collection

Each floating disk exposes a **statistics** sub-collection for disk-specific statistics. Each **statistic** contains the following elements:

Table 16.2. Elements for virtual disk statistics

Element	Type	Description
name	string	The unique identifier for the statistic entry.
description	string	A plain text description of the statistic.
unit	string	The unit or rate to measure the statistical values.
type	One of **GAUGE** or **COUNTER**	The type of statistic measures.
values type=	One of **INTEGER** or **DECIMAL**	The data type for the statistical values that follow.
value	complex	A data set that contains **datum**.
datum	see **values type**	An individual piece of data from a **value**.
disk id=	relationship	A relationship to the containing **disk** resource.

The following table lists the statistic types for floating disks.

Table 16.3. Disk statistic types

Name	Description
data.current.read	The data transfer rate in bytes per second when reading from the disk.
data.current.write	The data transfer rate in bytes per second when writing to the disk.

Example 16.3. An XML representation of a virtual machine's statistics sub-collection

```
<statistics>
    <statistic id="33b9212b-f9cb-3fd0-b364-248fb61e1272"
      href="/ovirt-engine/api/disks/f28ec14c-fc85-43e1-818d-
96b49d50e27b/statistics/
      33b9212b-f9cb-3fd0-b364-248fb61e1272">
        <name>data.current.read</name>
        <description>Read data rate</description>
        <values type="DECIMAL">
            <value>
                <datum>0</datum>
```

```
            </value>
        </values>
        <type>GAUGE</type>
        <unit>BYTES_PER_SECOND</unit>
        <disk id="f28ec14c-fc85-43e1-818d-96b49d50e27b"
          href="/ovirt-engine/api/disks/f28ec14c-fc85-43e1-818d-
96b49d50e27b"/>
    </statistic>
    ...
</statistics>
```

NOTE

This **statistics** sub-collection is read-only.

16.5. ACTIONS

16.5.1. Copying a Floating Disk

When copying a floating disk, the API requires the **storage_domain** element. The optional **name** element specifies an alias for the disk.

Example 16.4. Copying a Floating Disk

```
POST /ovirt-engine/api/disks/54a81464-b758-495a-824b-1e7937116ae5/copy
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain id="c8e108f7-c049-40d2-ad3d-620e4638828e"/>
    <disk>
        <name>rhel_disk2</name>
    </disk>
</action>
```

CHAPTER 17. TEMPLATES

17.1. VIRTUAL MACHINE TEMPLATE ELEMENTS

The **templates** collection provides information about the virtual machine templates in a Red Hat Virtualization environment. An API user accesses this information through the **rel="templates"** link obtained from the entry point URI.

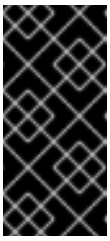
Additional information can be retrieved for **GET** requests using the **All-Content: true** header.

The following table shows specific elements contained in a virtual machine template resource representation.

Table 17.1. Virtual machine template elements

Element	Type	Description	Properties
link rel="disks"	relationship	A link to the **disks** sub-collection for virtual machine template resources.	
link rel="nics"	relationship	A link to the **nics** sub-collection for virtual machine template resources.	
link rel="cdroms"	relationship	A link to the **cdroms** sub-collection for virtual machine template resources.	
link rel="permissions"	relationship	A link to the **permissions** sub-collection for virtual machine template permissions.	
type	enumerated	The type of virtual machine the template provides. A list of enumerated values are available in **capabilities**.	
status	One of **illegal**, **locked** or **ok**	The template status. These states are listed in **template_states** under **capabilities**.	
memory	integer	The amount of memory allocated to the guest, in bytes.	
cpu	complex	The CPU **topology** (i.e. number of **sockets** and **cores**) available to the guest.	
os type=	string, e.g. **RHEL5** or **WindowsXP**	The guest operating system type.	

Element	Type	Description	Properties
os boot dev=	enumerated	A list of boot devices, described by a **dev** attribute on a **boot** element. A list of enumerated values are available in **capabilities**.	
os kernel	string	A path to a kernel image which the template is configured to boot from.	
os initrd	string	A path to an initrd image to be used with the kernel above.	
os cmdline	string	A kernel command line parameter string to be used with the kernel above.	
cluster id=	GUID	A reference to the template's host cluster.	
vm id=	GUID	A reference to the VM on which this template is based.	 
domain id=	GUID	A reference to the template's domain.	
creation_time	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The date and time at which this template was created.	
origin	One of **rhev**, **ovirt**, **vmware** or **xen**	The system from which this template originated.	
high_availability	complex	Set **enabled** to **true** if the VM should be automatically restarted if the host crashes. A **priority** element controls the order in which VMs are re-started.	
display	complex	The display **type** (either **vnc** or **spice**), port, and the number of **monitors**. The **allow_reconnect** Boolean value specifies if a client can reconnect to the machine via display.	

Element	Type	Description	Properties
stateless	Boolean: true or false	A stateless template contains a snapshot of its disk image taken at boot and deleted at shutdown. This means state changes do not persist after a reboot.	
usb	complex	Defines the USB policy for a virtual machine template. Requires an **enabled** element set to a Boolean value and a **type** element set to either **native** or **legacy**.  **IMPORTANT** The Legacy USB option has been deprecated and will be removed in Red Hat Virtualization 4.1.	
placement_policy	complex	Sets the placement policy for virtual machine migration. Requires a default **host=** and an **affinity** (one of **migratable**, **user_migratable** or **pinned**). Leave the **host** element empty to set no preferred host.	
custom_properties	complex	A set of user-defined environment variable passed as parameters to custom scripts. Each **custom_property** contains **name** and **value** attributes. A list of enumerated values are available in **capabilities**.	
timezone	tz database format: **Area/Location**	The the Sysprep timezone setting for a Windows virtual machine template.	
domain	complex	The the Sysprep domain setting for a Windows virtual machine template. Requires a **name** from the **domains** collection.	

17.2. XML REPRESENTATION OF A VIRTUAL MACHINE TEMPLATE

Example 17.1. An XML representation of a virtual machine template

```
<template href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000"
  id="00000000-0000-0000-0000-000000000000">
    <actions>
        <link href="/ovirt-engine/api/templates/00000000-0000-0000-
0000-000000000000/export"
          rel="export"/>
    </actions>
    <name>Blank</name>
    <description>Blank template</description>
    <comment>Blank template</comment>
    <link href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/disks"
      rel="disks"/>
    <link href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/nics"
      rel="nics"/>
    <link href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/cdroms"
      rel="cdroms"/>
    <link href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/permissions"
      rel="permissions"/>
    <link href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/watchdogs"
      rel="watchdogs"/>
    <type>server</type>
    <status>
        <state>ok</state>
    </status>
    <memory>536870912</memory>
    <cpu>
        <topology sockets="1" cores="1"/>
        <architecture>X86_64<architecture/>
    </cpu>
    <cpu_shares>0</cpu_shares>
    <os type="rhel_6x64">
        <boot dev="hd"/>
        <boot dev="cdrom"/>;
    </os>
    <cluster id="00000000-0000-0000-0000-000000000000"
      href="/ovirt-engine/api/clusters/00000000-0000-0000-0000-
000000000000"/>
    <creation_time>2010-08-16T14:24:29</creation_time>
    <origin>ovirt</origin>
    <high_availability>
        <enabled>true</enabled>
        <priority>100</priority>
    </high_availability>
    <display>
        <type>spice</type>
        <monitors>1</monitors>
        <single_qxl_pci>false</single_qxl_pci>
        <allow_override>true</allow_override>
        <smartcard_enabled>true</smartcard_enabled>
    </display>
```

```
    <stateless>false</stateless>
    <delete_protected>false</delete_protected>
    <sso>
      <methods>
        <method id="GUEST_AGENT">true</enabled>
      </methods>
    </sso>
    <usb>
        <enabled>true</enabled>
    </usb>
    <migration_downtime>-1</migration_downtime>
    <version>
      <base_template href="/ovirt-engine/api/templates/00000000-0000-
0000-0000-000000000000"
        id="00000000-0000-0000-0000-000000000000"/>
      <version_number>2</version_number>
      <version_name>RHEL65_TMPL_001</version_name>
    </version>
</template>
```

17.3. METHODS

17.3.1. Creating a New Template

Creation of a new template requires the **name** and **vm** elements. Identify the **vm** with the **id** attribute or **name** element.

Example 17.2. Creating a template from a virtual machine

```
POST /ovirt-engine/api/templates HTTP/1.1
Accept: application/xml
Content-type: application/xml

<template>
    <name>template1</name>
    <vm id="00000000-0000-0000-0000-000000000000"/>
</template>
```

17.3.2. Creating a New Template Sub Version

Creation of a new template sub version requires the **name** and **vm** elements for the new template, and the **base_template** and **version_name** elements for the new template version. The **base_template** and **version_name** elements must be specified within a **version** section enclosed in the **template** section. Identify the **vm** with the **id** attribute or **name** element.

Example 17.3. Creating a template sub version from a virtual machine

```
POST /ovirt-engine/api/templates HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

```
<template>
  <name>template1_001</name>
  <vm id="00000000-0000-0000-0000-000000000000"/>
  <version>
    <base_template id="00000000-0000-0000-0000-000000000000"/>
    <version_name>"template1_001"</version_name>
  </version>
</template>
```

17.3.3. Updating a Template

The **name**, **description**, **type**, **memory**, **cpu topology**, **os**, **high_availability**, **display**, **stateless**, **usb** and **timezone** elements can be updated after a template has been created.

Example 17.4. Updating a virtual machine template to contain 1 GB of memory

```
PUT /ovirt-engine/api/templates/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<template>
    <memory>1073741824</memory>
</template>
```

17.3.4. Updating a Template Sub Version

Only the **version_name** element can be updated after a template sub version has been created.

Example 17.5. Updating a virtual machine template sub version name

```
PUT /ovirt-engine/api/templates/00000000-0000-0000-0000-000000000000
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<template>
  <version>
    <version_name>template1_002</version_name>
  </version>
</template>
```

17.3.5. Removing a Template

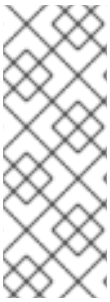
Removal of a virtual machine template requires a **DELETE** request.

Example 17.6. Removing a virtual machine template

```
DELETE /ovirt-engine/api/templates/00000000-0000-0000-0000-000000000000
HTTP/1.1

HTTP/1.1 204 No Content
```

17.4. ACTIONS

17.4.1. Export Template Action



NOTE

The export storage domain is deprecated. Storage data domains can be unattached from a data center and imported to another data center in the same environment, or in a different environment. Virtual machines, floating virtual disk images, and templates can then be uploaded from the imported storage domain to the attached data center. See the Importing Existing Storage Domains section in the *Red Hat Virtualization Administration Guide* for information on importing storage domains.

The **`templates`** collection contains an **`export`** action.

The export action exports a template to an **`Export`** storage domain. A destination storage domain is specified with a **`storage_domain`** reference.

The export action reports a failed action if a virtual machine template of the same name exists in the destination domain. Set the **`exclusive`** parameter to **`true`** to change this behavior and overwrite any existing virtual machine template.

Example 17.7. Action to export a template to an export storage domain

```
POST /ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000/export HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <storage_domain id="00000000-0000-0000-0000-000000000000"/>
    <exclusive>true<exclusive/>
</action>
```

CHAPTER 18. VIRTUAL MACHINE POOLS

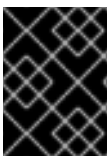
18.1. VIRTUAL MACHINE POOL ELEMENTS

The **vmpools** collection provides information about the virtual machine pools in a Red Hat Virtualization environment. An API user accesses this information through the **rel="vmpools"** link obtained from the entry point URI.

The following table shows specific elements contained in a virtual machine pool resource representation.

Table 18.1. Virtual machine pool elements

Element	Type	Description	Properties
name	string	A user-supplied, human readable name for the pool. The **name** is unique across all pool resources.	
description	string	A user-supplied, human readable description of the virtual machine pool.	
link rel="permissions"	relationship	A link to the **permissions** sub-collection for virtual machine pool permissions.	
size	integer	The number of virtual machines in the pool.	
cluster id=	GUID	A reference to the cluster resource in which virtual machines in this pool run.	
template id=	GUID	A reference to the template resource on which virtual machines in this pool are based.	
prestarted_vms	integer	The number of prestarted virtual machines in the virtual machine pool.	
max_user_vms	integer	The maximum number of virtual machines any one user can take from the virtual machine pool.	



IMPORTANT

The API as documented in this chapter is experimental and subject to change. It is not covered by the backwards compatibility statement.

18.2. XML REPRESENTATION OF A VIRTUAL MACHINE POOL

Example 18.1. An XML representation of a virtual machine pool

```
<vmpool href="/ovirt-engine/api/vmpools/2d2d5e26-1b6e-11e1-8cda-
001320f76e8e">
  id="2d2d5e26-1b6e-11e1-8cda-001320f76e8e"
    <actions>
     <link href="/ovirt-engine/api/vmpools/2d2d5e26-1b6e-11e1-8cda-
001320f76e8e/allocatevm"
       rel="allocatevm"/>
    </actions>
    <name>VMPool1</name>
    <description>Virtual Machine Pool 1</description>
    <size>2</size>
    <cluster href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95"/>
      id="99408929-82cf-4dc7-a532-9d998063fa95"
    <template href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000"/>
      id="00000000-0000-0000-0000-000000000000"
    <prestarted_vms>0</prestarted_vms>
    <max_user_vms>1</max_user_vms>
</vmpool>
```

18.3. METHODS

18.3.1. Creating a New Virtual Machine Pool

A new pool requires the **name**, **cluster** and **template** elements. Identify the **cluster** and **template** with the **id** attribute or **name** element.

Example 18.2. Creating a virtual machine pool

```
POST /ovirt-engine/api/vmpools HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vmpool>
    <name>VM_Pool_A</name>
    <cluster href="/ovirt-engine/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95"/>
      id="99408929-82cf-4dc7-a532-9d998063fa95"
    <template href="/ovirt-engine/api/templates/00000000-0000-0000-0000-
000000000000"/>
      id="00000000-0000-0000-0000-000000000000"
</vmpool>
```

18.3.2. Updating a Virtual Machine Pool

The **name**, **description**, **size**, **prestarted_vms** and **max_user_vms** can be updated after the virtual machine has been created.

Example 18.3. Updating a virtual machine pool

```
PUT /ovirt-engine/api/vmpools/2d2d5e26-1b6e-11e1-8cda-001320f76e8e
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vmpool>
    <name>VM_Pool_B</name>
    <description>Virtual Machine Pool B</description>
    <size>3</size>
    <prestarted_vms>1</size>
    <max_user_vms>2</size>
</vmpool>
```

18.3.3. Removing a Virtual Machine Pool

Removal of a virtual machine pool requires a **DELETE** request.

Example 18.4. Removing a virtual machine

```
DELETE /ovirt-engine/api/vmpools/2d2d5e26-1b6e-11e1-8cda-001320f76e8e
HTTP/1.1

HTTP/1.1 204 No Content
```

18.4. ACTIONS

18.4.1. Allocate Virtual Machine Action

The allocate virtual machine action allocates a virtual machine in the virtual machine pool.

Example 18.5. Action to allocate a virtual machine from a virtual machine pool

```
POST /ovirt-engine/api/vmpools/2d2d5e26-1b6e-11e1-8cda-
001320f76e8e/allocatevm HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

CHAPTER 19. DOMAINS

19.1. DOMAIN ELEMENTS

The API provides the ability to access user and group information from the organization's directory service using the **domains** collection. Domain information is referenced with the **rel="domains"** link.

Table 19.1. Domain elements

Element	Type	Description
name	string	The domain name.
link rel="users"	relationship	A link to the sub-collection for users associated with this domain.
link rel="groups"	relationship	A link to the sub-collection for groups associated with this domain.

The links to **users** and **groups** sub-collections also accept search queries.

NOTE

The **domains** collection and its sub-collections are read-only.

19.2. XML REPRESENTATION OF A DOMAIN RESOURCE

Example 19.1. An XML representation of a domain resource

```
<domain id="77696e32-6b38-7268-6576-2e656e676c61"
  href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-2e656e676c61">
    <name>domain.example.com</name>
    <link rel="users"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/users"/>
    <link rel="groups"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/groups"/>
    <link rel="users/search"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/
      users?search={query}"/>
    <link rel="groups/search"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/
      groups?search={query}"/>
</domain>
```

19.3. SUB-COLLECTIONS

19.3.1. Domain Users Sub-Collection

The **users** sub-collection contains all users in the directory service. This information is used to add new users to the Red Hat Virtualization environment.

Table 19.2. Domain user elements

Element	Type	Description
name	string	The name of the user.
last_name	string	The surname of the user.
user_name	string	The user name from directory service.
domain id	GUID	The containing directory service domain.
groups	complex	A list of directory service groups for this user.

Example 19.2. An XML representation of a user in the users sub-collection

```
<user id="225f15cd-e891-434d-8262-a66808fcb9b1"
  href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/users/
  d3b4e7be-5f57-4dac-b937-21e1771a501f">
    <name>RHEV-M Admin</name>
    <user_name>rhevmadmin@domain.example.com</user_name>
    <domain id="77696e32-6b38-7268-6576-2e656e676c61"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61"/>
    <groups>
        <group>
            <name>domain.example.com/Users/Enterprise Admins</name>
        </group>
        <group>
            <name>domain.example.com/Users/Domain Admins</name>
        </group>
        ...
    </groups>
</user>
```

19.3.2. Domain Groups Sub-Collection

The **groups** sub-collection contains all groups in the directory service. A domain **group** resource contains a set of elements.

Table 19.3. Domain group elements

Element	Type	Description
name	string	The name of the group.
domain id	GUID	The containing directory service domain.

Example 19.3. An XML representation of a group in the groups sub-collection

```
<group id="85bf8d97-273c-4a5c-b801-b17d58330dab"
  href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61/groups/
  85bf8d97-273c-4a5c-b801-b17d58330dab">
    <name>example.com/Users/Enterprise Admins</name>
    <domain id="77696e32-6b38-7268-6576-2e656e676c61"
      href="/ovirt-engine/api/domains/77696e32-6b38-7268-6576-
2e656e676c61"/>
</group>
```

CHAPTER 20. GROUPS

20.1. IMPORTED GROUP ELEMENTS

The **groups** collection contains imported groups from directory services. A **group** resource contains a set of elements.

Table 20.1. Imported group elements

Element	Type	Description
link rel="tags"	relationship	A link to the **tags** sub-collection for tags attached to this group.
link rel="permissions"	relationship	A link to the **permissions** sub-collection for permissions attached to this group.
link rel="roles"	relationship	A link to the **roles** sub-collection for roles attached to this group.

20.2. XML REPRESENTATION OF A GROUP RESOURCE

Example 20.1. An XML representation of a group resource

```
<group id="85bf8d97-273c-4a5c-b801-b17d58330dab"
  href="/ovirt-engine/api/groups/85bf8d97-273c-4a5c-b801-b17d58330dab">
    <name>Everyone</name>
    <link rel="tags"
      href="/ovirt-engine/api/groups/85bf8d97-273c-4a5c-b801-
b17d58330dab/tags"/>
    <link rel="permissions"
      href="/ovirt-engine/api/groups/85bf8d97-273c-4a5c-b801-
b17d58330dab/permissions"/>
    <link rel="roles"
      href="/ovirt-engine/api/groups/85bf8d97-273c-4a5c-b801-
b17d58330dab/roles"/>
    <domain_entry_id>
          656565303030302D303030302D303030302D303030
    </domain_entry_id>
    <namespace>*</namespace>
</group>
```

20.3. ADDING A GROUP FROM A DIRECTORY SERVICE

The API adds existing directory service groups to the Red Hat Virtualization Manager database with a **POST** request to the **groups** collection.

Example 20.2. Adding a group from a directory service

```
POST /ovirt-engine/api/group HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<group>
    <name>www.example.com/accounts/groups/mygroup</name>
    <domain>
     <name>example.com</name>
    </domain>
</group>
```

CHAPTER 21. ROLES

21.1. ROLE ELEMENTS

The **rel="roles"** link obtained from the entry point URI provides access to a static set of system roles. Each individual **role** element contains the following:

Table 21.1. Role elements

Element	Type	Description	Properties
link="permits"	relationship	A link to the **permits** sub-collection for role permits.	
mutable	Boolean: true or false	Defines the ability to update or delete the role. Roles with **mutable** set to **false** are roles built into the Red Hat Virtualization environment.	
administrative	Boolean: true or false	Defines the role as administrative-only.	

21.2. XML REPRESENTATION OF THE ROLES COLLECTION

Example 21.1. An XML representation of the roles collection

```
<roles>
    <role id="00000000-0000-0000-0000-000000000001"
      href="/ovirt-engine/api/roles/00000000-0000-0000-0000-
000000000001">
        <name>SuperUser</name>
        <description>Roles management administrator</description>
        <link rel="permits"
          href="/ovirt-engine/api/roles/00000000-0000-0000-0000-
000000000001/permits"/>
        <mutable>false</mutable>
        <administrative>true</administrative>
    </role>
    <role id="00000000-0000-0000-0001-000000000001"
      href="/ovirt-engine/api/roles/00000000-0000-0000-0001-
000000000001">
        <name>RHEVMUser</name>
        <description>RHEVM user</description>
        <link rel="permits"
          href="/ovirt-engine/api/roles/00000000-0000-0000-0001-
000000000001/permits"/>
        <mutable>false</mutable>
        <administrative>false</administrative>
    </role>
    <role id="00000000-0000-0000-0001-000000000002"
       href="/ovirt-engine/api/roles/00000000-0000-0000-0001-
000000000002">
        <name>RHEVMPowerUser</name>
```

```
        <description>RHEVM power user</description>
        <link rel="permits"
          href="/ovirt-engine/api/roles/00000000-0000-0000-0001-
000000000002/permits"/>
        <mutable>false</mutable>
        <administrative>false</administrative>
    </role>
</roles>
```

21.3. METHODS

21.3.1. Creating a Role

Creation of a role requires values for **name**, **administrative** and a list of initial **permits**.

Example 21.2. Creating a role

```
POST /ovirt-engine/api/roles HTTP/1.1
Accept: application/xml
Content-type: application/xml

<role>
    <name>Finance Role</name>
    <administrative>true</administrative>
    <permits>
        <permit id="1"/>
    </permits>
</role>
```

21.3.2. Updating a Role

The **name**, **description** and **administrative** elements are updatable post-creation.

Example 21.3. Updating a role

```
PUT /ovirt-engine/api/roles/8de42ad7-f307-408b-80e8-9d28b85adfd7
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<role>
    <name>Engineering Role</name>
    <description>Standard users in the Engineering Role</description>
    <administrative>false</administrative>
</role>
```

21.3.3. Removing a Role

Removal of a role requires a **DELETE** request.

Example 21.4. Removing a role

```
DELETE /ovirt-engine/api/roles/8de42ad7-f307-408b-80e8-9d28b85adfd7
HTTP/1.1

HTTP/1.1 204 No Content
```

21.4. ROLES PERMITS SUB-COLLECTION

21.4.1. Roles Permits Sub-Collection

Each role contains a set of allowable actions, or **permits**, which the API lists in **capabilities**.

A role's **permits** are listed as a sub-collection:

Example 21.5. Listing a role's permits

```
GET /ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9/permits
HTTP/1.1
Accept: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<permits>
    <permit id="1"
      href="/ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9/permits/1">
        <name>create_vm</name>
        <administrative>false</administrative>
        <role id="b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"
          href="/ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9"/>
    </permit>
    ...
</permits>
```

21.4.2. Assign a Permit to a Role

Assign a **permit** to a role with a **POST** request to the **permits** sub-collection. Use either an **id** attribute or a **name** element to specify the **permit** to assign.

Example 21.6. Assign a permit to a role

```
POST /ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9/permits HTTP/1.1
Accept: application/xml
Content-Type: application/xml
```

```
<permit id="1"/>

HTTP/1.1 201 Created
Content-Type: application/xml

<permits>
    <permit id="1"
      href="/ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9/permits/1">
        <name>create_vm</name>
        <administrative>false</administrative>
        <role id="b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"
          href="/ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9"/>
    </permit>
</permits>
```

21.4.3. Remove a Permit from a Role

Remove a **permit** from a role with a **DELETE** request to the **permit** resource.

Example 21.7. Remove a permit from a role

```
DELETE /ovirt-engine/api/roles/b67dfbe2-0dbc-41e4-86d3-
a2fbef02cfa9/permits/1 HTTP/1.1

HTTP/1.1 204 No Content
```

CHAPTER 22. USERS

22.1. USER ELEMENTS

Users are exposed in a top-level collection and are referenced with the **rel="users"** link. Individual **user** elements contain the following:

Table 22.1. User elements

Element	Type	Description	Properties
user_name	string	The user principal name (UPN). The UPN is used as a more convenient identifier when adding a new user.	
link rel="tags"	relationship	A link to the **tags** sub-collection for user resources.	
link rel="roles"	relationship	A link to the **roles** sub-collection for user resources.	
name	string	A free-text name for the user.	
domain	string	The containing directory service domain.	
groups	complex	A list of directory service groups for this user.	

22.2. XML REPRESENTATION OF A USER RESOURCE

Example 22.1. An XML representation of a user resource

```
GET /ovirt-engine/api/users HTTP/1.1
Accept: application/xml

<user id="225f15cd-e891-434d-8262-a66808fcb9b1"
  href="/ovirt-engine/api/users/225f15cd-e891-434d-8262-a66808fcb9b1">
    <name>RHEV-M Admin</name>
    <actions/>
    <link rel="roles"
      href="/ovirt-engine/api/users/225f15cd-e891-434d-8262-
a66808fcb9b1/roles"/>
    <link rel="tags"
      href="/ovirt-engine/api/users/225f15cd-e891-434d-8262-
a66808fcb9b1/tags"/>
    <domain>domain.example.com</domain>
    <logged_in>false</logged_in>
```

```
    <user_name>rhevmadmin@domain.example.com</user_name>
    <groups>
        <group>Group Policy Creator
Owners@domain.example.com/Users</group>
        <group>Domain Admins@domain.example.com/Users</group>
        <group>Enterprise Admins@domain.example.com/Users</group>
        <group>Schema Admins@domain.example.com/Users</group>
        <group>Administrators@domain.example.com/Builtin</group>
    </groups>
</user>
```

22.3. METHODS

22.3.1. Adding a User

The API adds an existing directory service user to the Red Hat Virtualization Manager database with a **POST** request to the **users** collection. The client-provided new user representation includes an embedded **roles** list with at least one initial **role** to assign to the user. For example, the following request assigns two initial roles to the user **joe@domain.example.com**:

Example 22.2. Adding a user from directory service and assigning two roles

```
POST /ovirt-engine/api/users HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<user>
    <user_name>joe@domain.example.com</user_name>
    <roles>
        <role>
            <name>RHEVMPowerUser</name>
        </role>
        <role id="00000000-0000-0000-0001-000000000003"/>
    </roles>
</user>
```

The new user is identified either by Red Hat Virtualization Manager user ID or via the directory service user principal name (UPN). The user ID format reported from the directory service domain might be different to the expected Red Hat Virtualization Manager format, such as in LDIF [5] , the ID has the opposite byte order and is base-64 encoded. Hence it is usually more convenient to refer to the new user by UPN.

NOTE

The user exists in the directory service domain before it is added to the Red Hat Virtualization Manager database. An API user has the option to query this domain through the **domains** collection prior to creation of the user.

Roles are identified either by name or ID. The example above shows both approaches.

22.3.2. Adding Roles to a User

Further roles are attached or detached with **POST** or **DELETE** requests to the roles sub-collection of an individual user. The example below illustrates how the API adds the **RHEVMVDIUser** role to the role assignments for a particular user.

NOTE

The embedded user roles list of the **user** element is only used for the initial creation. All interactions post-creation with the user's role assignments go through the **roles** sub-collection.

Example 22.3. Adding roles to a user

```
POST /ovirt-engine/api/users/225f15cd-e891-434d-8262-a66808fcb9b1/roles
HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<role>
    <name>RHEVMVDIUser</name>
</role>
```

[5] The LDAP Data Interchange Format is described in RFC 2849.

CHAPTER 23. MAC ADDRESS POOLS

23.1. MAC ADDRESS POOL ELEMENTS

The **macpools** collection provides information about the MAC address pools in a Red Hat Virtualization environment. An API user accesses this information through the **rel="macpools"** link obtained from the entry point URI. The following table shows specific elements contained in a MAC address pool resource representation.

Table 23.1. MAC address pool elements

Element	Type	Description	Properties
name	string	A plain text, human-readable name for the MAC address pool.	
description	string	A plain text, human-readable description of the MAC address pool.	
allow_duplicates	Boolean: true or false	Defines whether duplicate MAC addresses are permitted in the pool. If not specified, **allow_duplicates** defaults to false.	
default_pool	Boolean: true or false	Defines whether this is the default pool. If not specified, **default_pool** defaults to false.	
ranges	complex	Defines the range of MAC addresses for the pool. Multiple ranges can be defined within the **ranges** element.	

23.2. XML REPRESENTATION OF THE MAC ADDRESS POOLS COLLECTION

Example 23.1. An XML representation of the MAC address pools collection

```
<mac_pools>
    <mac_pool href="/ovirt-engine/api/macpools/00000000-0000-0000-0000-
000000000000" id="00000000-0000-0000-0000-000000000000">
        <name>Default</name>
        <description>Default MAC pool</description>
        <allow_duplicates>false</allow_duplicates>
        <default_pool>true</default_pool>
        <ranges>
            <range>
                <from>00:1A:4A:16:01:51</from>
                <to>00:1A:4A:16:01:e6</to>
            </range>
        </ranges>
    </mac_pool>
```

```
</mac_pools>
```

23.3. METHODS

23.3.1. Creating a MAC Address Pool

Creation of a MAC address pool requires values for **name** and **ranges**.

Example 23.2. Creating a MAC address pool

```
POST /ovirt-engine/api/macpools HTTP/1.1
Accept: application/xml
Content-type: application/xml

<mac_pool>
    <name>MACPool</name>
    <description>A MAC address pool</description>
    <allow_duplicates>true</allow_duplicates>
    <default_pool>false</default_pool>
    <ranges>
        <range>
            <from>00:1A:4A:16:01:51</from>
            <to>00:1A:4A:16:01:e6</to>
        </range>
    </ranges>
</mac_pool>
```

23.3.2. Updating a MAC Address Pool

The **name**, **description**, **allow_duplicates**, and **ranges** elements are updatable post-creation.

Example 23.3. Updating a MAC address pool

```
PUT /ovirt-engine/api/macpools/ab39bbc1-1d64-4737-9b20-ce081f99b0e1
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<mac_pool>
    <name>UpdatedMACPool</name>
    <description>An updated MAC address pool</description>
    <allow_duplicates>false</allow_duplicates>
    <ranges>
        <range>
            <from>00:1A:4A:16:01:51</from>
            <to>00:1A:4A:16:01:e6</to>
        </range>
        <range>
            <from>02:1A:4A:01:00:00</from>
            <to>02:1A:4A:FF:FF:FF</to>
        </range>
```

```
    </ranges>
</mac_pool>
```

23.3.3. Removing a MAC Address Pool

Removal of a MAC address pool requires a **DELETE** request.

Example 23.4. Removing a MAC address pool

```
DELETE /ovirt-engine/api/macpools/ab39bbc1-1d64-4737-9b20-ce081f99b0e1
HTTP/1.1

HTTP/1.1 204 No Content
```

CHAPTER 24. TAGS

24.1. TAG ELEMENTS

The **tags** collection provides information about tags in a Red Hat Virtualization environment. An API user accesses this information through the **rel="tags"** link obtained from the entry point URI.

The following table shows specific elements contained in a tag resource representation.

Table 24.1. Tag elements

Element	Type	Description	Properties
host	GUID	A reference to the host which the tag is attached.	
user	GUID	A reference to the user which the tag is attached.	
vm	GUID	A reference to the VM which the tag is attached.	
parent	complex	A reference to the VM which the tag is attached.	

24.2. XML REPRESENTATION OF A TAG RESOURCE

Example 24.1. An XML representation of a tag resource

```
<tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"
  href="/ovirt-engine/api/tags/f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e">
    <name>Finance</name>
    <description>Resources for the Finance department</description>
    <parent>
        <tag id="-1" href="/ovirt-engine/api/tags/-1"/>
    </parent>
</tag>
```

24.3. ASSOCIATING TAGS

24.3.1. Associating Tags With a Host, User or VM

The collection referenced by **link rel="tags"** from a **host**, **user** or **vms** represents the set of tags associated with the entity.

These **tag** representations also contain a **host id**, **user id** or **vm id** reference to the entity in question.

Associating a tag with an entity is achieved by **POST**ing a tag reference (identifying the tag either by its **id** or **name**) to the collection.

Example 24.2. Associating a tag with a virtual machine

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/tags
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<tag>
    <name>Finance</name>
</tag>

HTTP/1.1 201 Created
Content-Type: application/xml

<tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"
  href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/tags/
  f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e">
    <name>Finance</name>
    <description>Resources for the Finance department</description>
    <vm id="5114bb3e-a4e6-44b2-b783-b3eea7d84720"
      href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720"/>
</tag>
```

24.3.2. Removing a Tag

Removing an association is achieved with a **DELETE** request to the appropriate element in the collection.

Example 24.3. Removing a tag from a virtual machine

```
DELETE /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-
b3eea7d84720/tags/f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e HTTP/1.1

HTTP/1.1 204 No Content
```

24.3.3. Querying a Collection for Tagged Resources

To query the set of entities associated with a given tag, the **collection/search** URI template for the appropriate collection should be used to search for entities matching **tag=MyTag**.

Example 24.4. Querying a collection for tagged resources

```
GET /ovirt-engine/api/vms?search=tag%3DFinance HTTP/1.1
Accept: application/xml

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

```
<vms>
    <vm id="5114bb3e-a4e6-44b2-b783-b3eea7d84720"
      href="/ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720">
        ...
    </vm>
    ...
</vms>
```

24.4. PARENT TAGS

24.4.1. Parent Tags

An API user assigns a **parent** element to a tag to create a hierarchical link to a parent tag. The tags are presented as a flat collection, which descends from the **root** tag, with tag representations containing a link element to a parent tag

NOTE

The **root** tag is a special pseudo-tag assumed as the default parent tag if no parent tag is specified. The **root** tag cannot be deleted nor assigned a parent tag.

This tag hierarchy is expressed in the following way:

Example 24.5. Tag Hierarchy

```
<tags>
    <tag id="-1" href="/ovirt-engine/api/tags/-1">
        <name>root</name>
        <description>root</description>
        <parent>
            <tag id="-1" href="/ovirt-engine/api/tags/-1"/>
        </parent>
    </tag>
    <tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"
      href="/ovirt-engine/api/tags/f436ebfc-67f2-41bd-8ec6-
902b6f7dcb5e">
        <name>Finance</name>
        <description>Resources for the Finance department</description>
        <parent>
            <tag id="-1" href="/ovirt-engine/api/tags/-1"/>
        </parent>
    </tag>
    <tag id="ac18dabf-23e5-12be-a383-a38b165ca7bd"
      href="/ovirt-engine/api/tags/ac18dabf-23e5-12be-a383-
a38b165ca7bd">
        <name>Billing</name>
        <description>Billing Resources</description>
        <parent>
            <tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"
              href="/ovirt-engine/api/tags/f436ebfc-67f2-41bd-8ec6-
902b6f7dcb5e"/>
```

```
        </parent>
    </tag>
</tags>
```

In this XML representation, the tags follow this hierarchy:

```
root              (id: -1)
  - Finance       (id: f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e)
      - Billing   (id: ac18dabf-23e5-12be-a383-a38b165ca7bd)
```

24.4.2. Setting a Parent Tag

POSTing a new tag with a **parent** element creates an association with a parent tag, using either the **id** attribute or the **name** element to reference the parent tag

Example 24.6. Setting an association with a parent tag with the id attribute

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/tags
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<tag>
    <name>Billing</name>
    <description>Billing Resources</description>
    <parent>
        <tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"/>
    </parent>
</tag>
```

Example 24.7. Setting an association with a parent tag with the name element

```
POST /ovirt-engine/api/vms/5114bb3e-a4e6-44b2-b783-b3eea7d84720/tags
HTTP/1.1
Accept: application/xml
Content-Type: application/xml

HTTP/1.1 200 OK
Content-Type: application/xml

<tag>
    <name>Billing</name>
    <description>Billing Resources</description>
    <parent>
        <tag>
            <name>Finance</name>
```

```
        </tag>
    </parent>
</tag>
```

24.4.3. Changing a Parent Tag

A tag changes a parent using a **PUT** request:

Example 24.8. Changing the parent tag

```
PUT /ovirt-engine/api/tags/ac18dabf-23e5-12be-a383-a38b165ca7bd HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<tag>
    <parent>
        <tag id="f436ebfc-67f2-41bd-8ec6-902b6f7dcb5e"/>
    </parent>
</tag>
```

CHAPTER 25. EVENTS

25.1. EVENT ELEMENTS

The **rel="events"** link obtained from the entry point URI accesses the **events** collection and lists system events from Red Hat Virtualization Manager.

Table 25.1. Event elements

Element	Type	Description
description	string	A description of the system event
code	integer	The integer event code.
severity	One of **normal**, **warning**, **error** or **alert**	The level of severity for the event.
time	**xsd:dateTime** format: **YYYY-MM-DDThh:mm:ss**	The timestamp indicating when the event happened.
correlation_id	string	The identification string for an action that is spread across layers of Red Hat Virtualization.
user id=	GUID	The identification code for the user who triggered the event.
origin	string	The source of the event. Standard events are reported by **oVirt**.
custom_id	integer	A custom identification number for custom events. Standard events have a **custom_id** of **-1**.
flood_rate	integer	The time, in seconds, during which the same event cannot reoccur in the event list. The default value is **30**.
external_status	complex	The external health status of a host. Contains the **state** element, which can be one of **ok**, **info**, **error**, **warning**, or **failure**.

25.2. XML REPRESENTATION OF THE EVENTS COLLECTION

Example 25.1. An XML representation of the events collection

```
<events>
    <event id="537" href="/ovirt-engine/api/events/537">
```

```
        <description>User vdcadmin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2011-01-12T10:48:27.827+02:00</time>
        <user id="9b9002d1-ec33-4083-8a7b-31f6b8931648"
          href="/ovirt-engine/api/users/9b9002d1-ec33-4083-8a7b-
31f6b8931648"/>
    </event>
    ...
</events>
```

25.3. XML REPRESENTATION OF A VIRTUAL MACHINE CREATION EVENT

In addition to **user**, an **event** representation also contains a set of XML element relationships to resources relevant to the event.

Example 25.2. An XML representation of a virtual machine creation event

```
<event id="635" href="/ovirt-engine/api/events/635">
    <description>VM bar was created by rhevadmin.</description>
    <code>34</code>
    <severity>normal</severity>
    <time>2011-07-11T16:32:03.172+02:00</time>
    <user id="4621b611-43eb-4d2b-ae5f-1180850268c4"
      href="/ovirt-engine/api/users/4621b611-43eb-4d2b-ae5f-
1180850268c4"/>
    <vm id="9b22d423-e16b-4dd8-9c06-c8e9358fbc66"
      href="/ovirt-engine/api/vms/9b22d423-e16b-4dd8-9c06-
c8e9358fbc66"/>
    <storage_domain id="a8a0e93d-c570-45ab-9cd6-3c68ab31221f"
      href="/ovirt-engine/api/storagedomains/a8a0e93d-c570-45ab-9cd6-
3c68ab31221f"/>
</event>
```

This example representation provides XML element relationships to a virtual machine resource and a storage domain resource.

25.4. METHODS

25.4.1. Searching Events

The **events** collection provides search queries similar to other resource collections. An additional feature when searching the **events** collection is the ability to search from a certain event. This queries all of events since a specified event.

Querying from an event requires an additional **from** parameter added before the search query. This **from** argument references an event **id** code.

Example 25.3. Searching from an event

```
GET /ovirt-engine/api/events;from=1012?search=type%3D30 HTTP/1.1
Accept: application/xml
```

This displays all events with **type** set to 30 since **id="1012"**

```
HTTP/1.1 200 OK
Content-Type: application/xml
<events>
    <event id="1018" href="/ovirt-engine/api/events/1018">
        <description>User admin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2011-07-11T14:03:22.485+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
    </event>
    <event id="1016" href="/ovirt-engine/api/events/1016">
        <description>User admin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2011-07-11T14:03:07.236+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
    </event>
    <event id="1014" href="/ovirt-engine/api/events/1014">
        <description>User admin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2011-07-11T14:02:16.009+10:00</time>
        <user id="80b71bae-98a1-11e0-8f20-525400866c73"
          href="/ovirt-engine/api/users/80b71bae-98a1-11e0-8f20-
525400866c73"/>
    </event>
</events>
```

Example 25.4. Searching using a specific event severity

```
GET /ovirt-engine/api/events?search=severity>normal HTTP/1.1
Accept: application/xml
```

This displays all events with severity higher than **normal**. Severity levels include **normal**, **warning**, **error** and **alert**.

```
HTTP/1.1 200 OK
Content-Type: application/xml
<events>
    <event id="2823" href="/ovirt-engine/api/events/2823">
        <description>Host Host-05 has time-drift of 36002 seconds while
maximum configured value is 300 seconds.</description>
        <code>604</code>
        <severity>warning</severity>
```

```
        <time>2015-07-11T14:03:22.485+10:00</time>
        <host href= "/ovirt-engine/api/hosts/44e52bb2-27d6-4d35-8038-
0c4b4db89789" id="44e52bb2-27d6-4d35-8038-0c4b4db89789"/>
        <cluster href= "/ovirt-engine/api/clusters/00000001-0001-0001-
0001-00000000021b" id="00000001-0001-0001-0001-00000000021b"/>
        <origin>oVirt</origin>
        <custom_id>-1</custom_id>
        <flood_rate>30</flood_rate>
    </event>
...
</events>
```

25.4.2. Paginating Events

A virtualization environment generates a large amount of events after a period of time. However, the API only displays a default number of events for one search query. To display more than the default, the API separates results into pages with the **page** command in a search query.

The following search query tells the API to paginate results using a **page** value in combination with the **sortby** clause:

```
sortby time asc page 1
```

The **sortby** clause defines the base element to order of the results and whether the results are ascending or descending. For search queries of **events**, set the base element to **time** and the order to ascending (**asc**) so the API displays all events from the creation of your virtualization environment.

The **page** condition defines the page number. One page equals the default number of events to list. Pagination begins at **page 1**. To view more pages, increase the **page** value:

```
sortby time asc page 2

sortby time asc page 3

sortby time asc page 4
```

Example 25.5. Paginating events

This example paginates **event** resources. The URL-encoded request is:

```
GET /ovirt-engine/api/events?search=sortby%20time%20asc%20page%201
HTTP/1.1
Accept: application/xml
```

Increase the **page** value to view the next page of results.

```
GET /ovirt-engine/api/events?search=sortby%20time%20asc%20page%202
HTTP/1.1
Accept: application/xml
```

Use an additional **from** argument to set the starting **id**.

```
GET /ovirt-engine/api/events?
```

```
search=sortby%20time%20asc%20page%202&from=30 HTTP/1.1
Accept: application/xml
```

25.4.3. Adding Events

The API can add custom events with a **POST** request to the **events** collection. A new event requires the **description**, **severity**, **origin**, and **custom_id** elements. Custom events can also include **flood_rate**, **user id**, and the **id** codes of any resources relevant to the event. **host** and **storage_domain** elements can contain the **external_status** element to set an external health status.

Example 25.6. Adding a custom event to the event list

```
POST /ovirt-engine/api/events HTTP/1.1
Accept: application/xml
Content-type: application/xml

<event>
  <description>The heat of the host is above 30 Oc</description>
  <severity>warning</severity>
  <origin>HP Openview</origin>
  <custom_id>1</custom_id>
  <flood_rate>30</flood_rate>
  <host id="f59a29cd-587d-48a3-b72a-db537eb21957" >
    <external_status>
       <state>warning</state>
    </external_status>
  </host>
</event>
```

25.4.4. Removing Events

Removal of an event from the event list requires a **DELETE** request.

Example 25.7. Removing an event

```
DELETE /ovirt-engine/api/events/1705 HTTP/1.1

HTTP/1.1 204 No Content
```

APPENDIX A. API USAGE WITH CURL

A.1. API USAGE WITH CURL

This appendix provides instructions on adapting REST requests for use with **cURL**. **cURL** is a command line tool for transferring data across various protocols, including HTTP, and supports multiple platforms such as Linux, Windows, Mac and Solaris. Most Linux distributions include **cURL** as a package.

A.2. INSTALLING CURL

A Red Hat Enterprise Linux user installs **cURL** with the following terminal command:

```
yum install curl
```

For other platforms, seek installation instructions on the **cURL** website (http://curl.haxx.se/).

A.3. USING CURL

cURL uses a command line interface to send requests to a HTTP server. Integrating a request requires the following command syntax:

```
Usage: curl [options] uri
```

The **uri** refers to target HTTP address to send the request. This is a location on your Red Hat Virtualization Manager host within the API entry point path (**/ovirt-engine/api**).

cURL options

-X *COMMAND*, --request *COMMAND*

The request command to use. In the context of the REST API, use **GET**, **POST**, **PUT** or **DELETE**.

Example: **-X GET**

-H *LINE*, --header *LINE*

HTTP header to include with the request. Use multiple header options if more than one header is required.

Example: **-H "Accept: application/xml" -H "Content-Type: application/xml"**

-u *USERNAME:PASSWORD*, --user *USERNAME:PASSWORD*

The user name and password of the Red Hat Virtualization user. This attribute acts as a convenient replacement for the **Authorization:** header.

Example: **-u admin@internal:p@55w0rd!**

--cacert *CERTIFICATE*

The location of the certificate file for SSL communication to the REST API. The certificate file is saved locally on the client machine. Use the **-k** attribute to bypass SSL.

Example: **--cacert ~/Certificates/rhevm.cer**

-d *BODY*, --data *BODY*

The body to send for requests. Use with **POST**, **PUT** and **DELETE** requests. Ensure to specify the **Content-Type: application/xml** header if a body exists in the request.

Example: **-d "<cdrom><file id='rhel-server-6.0-x86_64-dvd.iso'/></cdrom>"**

A.4. EXAMPLES

A.4.1. GET Request with cURL

Example A.1. GET request

The following **GET** request lists the virtual machines in the **vms** collection. Note that a **GET** request does not contain a body.

```
GET /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
```

Adapt the method (**GET**), header (**Accept: application/xml**) and URI (**https://*[RHEVM-Host]*:443/ovirt-engine/api/vms**) into the following **cURL** command:

```
$ curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert
[CERT] https://[RHEVM-Host]:443/ovirt-engine/api/vms
```

An XML representation of the **vms** collection displays.

A.4.2. POST Request with cURL

Example A.2. POST request

The following **POST** request creates a virtual machine in the **vms** collection. Note that a **POST** request requires a body.

```
POST /ovirt-engine/api/vms HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
  <name>vm1</name>
  <cluster>
    <name>default</name>
  </cluster>
  <template>
    <name>Blank</name>
  </template>
  <memory>536870912</memory>
  <os>
    <boot dev="hd"/>
  </os>
</vm>
```

Adapt the method (**POST**), headers (**Accept: application/xml** and **Content-type: application/xml**), URI (**https://*[RHEVM-Host]*:443/ovirt-engine/api/vms**) and request body into the following **cURL** command:

```
$ curl -X POST -H "Accept: application/xml" -H "Content-type:
application/xml" -u [USER:PASS] --cacert [CERT] -d "<vm>
<name>vm1</name><cluster><name>default</name></cluster><template>
<name>Blank</name></template><memory>536870912</memory><os><boot
dev='hd'/></os></vm>" https://[RHEVM-Host]:443/ovirt-engine/api/vms
```

The REST API creates a new virtual machine and displays an XML representation of the resource.

A.4.3. PUT Request with cURL

Example A.3. PUT request

The following **PUT** request updates the memory of a virtual machine resource. Note that a **PUT** request requires a body.

```
PUT /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<vm>
    <memory>1073741824</memory>
</vm>
```

Adapt the method (**PUT**), headers (**Accept: application/xml** and **Content-type: application/xml**), URI (**https://*[RHEVM-Host]*:443/ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399**) and request body into the following **cURL** command:

```
$ curl -X PUT -H "Accept: application/xml" -H "Content-type:
application/xml" -u [USER:PASS] --cacert [CERT] -d "<vm>
<memory>1073741824</memory></vm>" https://[RHEVM-Host]:443//ovirt-
engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c039
```

The REST API updates the virtual machine with a new memory configuration.

A.4.4. DELETE Request with cURL

Example A.4. DELETE request

The following **DELETE** request removes a virtual machine resource.

```
DELETE /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399
HTTP/1.1
```

Adapt the method (**DELETE**) and URI (**https://*[RHEVM-Host]*:443/ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399**) into the following **cURL** command:

```
$ curl -X DELETE -u [USER:PASS] --cacert [CERT] https://[RHEVM-
Host]:443//ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c039
```

The REST API removes the virtual machine. Note the **Accept: application/xml** request header is optional due to the empty result of **DELETE** requests.

A.4.5. DELETE Request Including Body with cURL

Example A.5. DELETE request with body

The following **DELETE** request force removes a virtual machine resource as indicated with the optional body.

```
DELETE /ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
  <force>true</force>
</action>
```

Adapt the method (**DELETE**), headers (**Accept: application/xml** and **Content-type: application/xml**), URI (**https://*[RHEVM-Host]*:443/ovirt-engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c0399**) and request body into the following **cURL** command:

```
$ curl -X DELETE -H "Accept: application/xml" -H "Content-type:
application/xml" -u [USER:PASS] --cacert [CERT] -d "<action>
<force>true</force></action>" https://[RHEVM-Host]:443//ovirt-
engine/api/vms/082c794b-771f-452f-83c9-b2b5a19c039
```

The REST API force removes the virtual machine.