Red Hat Storage 3
Console Developer Guide

Red Hat Storage Console Developer Guide Draft version

Anjana Suparna Sriram Pavithra Srinivasan Divya Muntimadugu

Red Hat Storage 3 Console Developer Guide

Red Hat Storage Console Developer Guide Draft version

Anjana Suparna Sriram
Red Hat Engineering Content Services
asriram@redhat.com

Pavithra Srinivasan
Red Hat Engineering Content Services
psriniva@redhat.com

Divya Muntimadugu
Red Hat Engineering Content Services
divya@redhat.com

Legal Notice

Abstract

Using the Red Hat Storage Console Application Programming Interfaces.

Table of Contents

Preface

Part I. Introduction

Chapter 1. Introduction

Red Hat Storage Console provides a **Representational State Transfer (REST)** API. The API provides software developers and system administrators with control over their Red Hat Storage 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 Storage environment with custom scripts or external applications that access the API through 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 storage infrastructure as many details are discovered at runtime;
- Resource-based model - The resource-based REST model provides a natural way to manage a storage platform.

This provides developers and administrators with the ability to:

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

This documentation acts as a reference to the Red Hat Storage Console REST API. It aims to provide developers and administrators with instructions and examples to help harness the functionality of their Red Hat Storage environment through the REST API.

Report a bug

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.

Report a bug

1.2. Prerequisites

This guide requires the following:

- An installation of Red Hat Storage Console, which includes the REST API;

- A client or programming library that initiates and receives HTTP requests from the REST API. As an example, this guide includes basic instructions on use with **cURL** in Appendix B, *API Usage with cURL*;
- 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; and,
- Knowledge of Extensible Markup Language (XML), which the API uses to construct resource representations. The W3C provides a full specification on XML at http://www.w3.org/TR/xml/.

Report a bug

Chapter 2. Authentication and Security

This chapter provides information on authorization through Red Hat Storage Console's security.

Report a bug

2.1. TLS/SSL Certification

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

Procedure 2.1. Attain a certificate

This process helps a user attain a certificate from the Red Hat Storage Console and transfer it to the client machine. A user achieves this using one of two methods:

1. **Method 1** - Use a command line tool to download the certificate from the server. Examples of command line tools include **cURL** and **Wget**; both are available for multiple platforms.

    a. If using **cURL**:

    ```
    curl -o rhsc.cer http://[rhsc-server]/ca.crt
    ```

    b. If using **Wget**:

    ```
    wget -O rhsc.cer http://[rhsc-server]/ca.crt
    ```

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

    ```
    http://[rhsc-server]/ca.crt
    ```

    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 **rhsc.cer**.

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

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

Procedure 2.2. Import a certificate to your client

- A certificate import for your client relies on how the client itself stores and interprets certificates. This guide contains an example on importing to a Java keystore in Appendix D, *Java Keystores*. For clients not using Network Security Services (NSS) or Java KeyStore (JKS), please refer to your client documentation for more information on importing a certificate.

Report a bug

2.2. HTTP Authentication

Any user with a Red Hat Storage Console account has access to the REST API. An API user submits a mandatory Red Hat Storage Console username and password with all requests to the API and uses HTTP Basic Authentication [1] 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 Storage Console 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 access to the API

Item	Value
username	rhscadmin
domain	domain.example.com
password	123456
unencoded credentials	rhscadmin@domain.example.com:123456
base64 encoded credentials	cmhzY2FkbWluQGRvbWFpbi5leGFtcGxlLmNvbToxMjM0NTYK

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 cmhzY2FkbWluQGRvbWFpbi5leGFtcGxlLmNvbToxMjM0NTYK

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.

Report a bug

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.

To request an authenticated session:

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

   ```
   HEAD [base] HTTP/1.1
   Host: [host]
   Authorization: Basic
   cmhzY2FkbWluQGRvbWFpbi5leGFtcGxlLmNvbToxMjM0NTYK
   Prefer: persistent-auth

   HTTP/1.1 200 OK
   ...
   ```

   This returns a response with the following header:

   ```
   Set-Cookie: JSESSIONID=5dQja5ubr4yvI2MM2z+LZxrK; Path=/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
   ```

```
cmhzY2FkbWluQGRvbWFpbi5leGFtcGxlLmNvbToxMjM0NTYK

HTTP/1.1 200 OK
...
```

Report a bug

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

Chapter 3. REST API Quick Start Examples

This chapter provides an example to demonstrate the REST API's ability to setup a basic trusted storage pool within the Red Hat Storage environment.

This example uses **cURL** to demonstrate REST requests with a client application. Find out more about **cURL** use with the REST API in Appendix B, *API Usage with cURL*. 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 Storage Console.

Important

All **cURL** examples include placeholders for authentication details (**USER:PASS**) and certificate location (**CERT**). Ensure all requests performed with **cURL** fulfil certification and authentication requirements. See Chapter 2, *Authentication and Security* and Appendix B, *API Usage with cURL* for more information.

Note

Red Hat Storage Console 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 Storage Console environment.

Report a bug

3.1. Example: Access API Entry Point

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

Example 3.1. Access the API entry point

```
GET /api HTTP/1.1
Accept: application/xml
```

cURL command:

```
curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert [CERT]
https://[RHSC Host]/api
```

The API returns the following representation:

```
HTTP/1.1 200 OK
```

```
Content-Type: application/xml

<api>
    <link href="/api/capabilities" rel="capabilities"/>
    <link href="/api/clusters" rel="clusters"/>
    <link href="/api/clusters?search={query}" rel="clusters/search"/>
    <link href="/api/events" rel="events"/>
    <link href="/api/events;from={event_id}?search={query}"
rel="events/search"/>
    <link href="/api/hosts" rel="hosts"/>
    <link href="/api/hosts?search={query}" rel="hosts/search"/>
    <link href="/api/networks" rel="networks"/>
    <link href="/api/networks?search={query}" rel="networks/search"/>
    <link href="/api/roles" rel="roles"/>
    <link href="/api/tags" rel="tags"/>
    <link href="/api/users" rel="users"/>
    <link href="/api/users?search={query}" rel="users/search"/>
    <link href="/api/groups" rel="groups"/>
    <link href="/api/groups?search={query}" rel="groups/search"/>
    <link href="/api/domains" rel="domains"/>
    <special_objects>
        <link href="/api/tags/00000000-0000-0000-0000-000000000000"
rel="tags/root"/>
    </special_objects>
    <product_info>
        <name>Red Hat Storage Console</name>
        <vendor>Red Hat</vendor>
        <version major="3" minor="0" build="0" revision="0"/>
        <full_version>3.0.0-0.10.el6_5</full_version>
    </product_info>
    <summary>
        <hosts>
            <total>0</total>
            <active>0</active>
        </hosts>
        <users>
            <total>1</total>
            <active>1</active>
        </users>
    </summary>
    <time>2014-06-23T06:47:18.567+05:30</time>
</api>
      <total>1</total>
      <active>1</active>
    </users>
  </summary>
</api>
```

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

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

Report a bug

3.2. Example: List Cluster Collection

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

The following request retrieves a representation of the cluster collection:

Example 3.2. List clusters collection

```
GET /api/clusters HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* --cacert *[CERT]* https://*[RHSC Host]*/api/clusters

The API returns the following representation:

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

<clusters>
    <cluster href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95"
id="99408929-82cf-4dc7-a532-9d998063fa95">
        <name>Default</name>
        <description>The default server cluster</description>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks" rel="networks"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/permissions" rel="permissions"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes" rel="glustervolumes"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glusterhooks" rel="glusterhooks"/>
        <data_center href="/api/datacenters/5849b030-626e-47cb-ad90-
3ce782d831b3" id="5849b030-626e-47cb-ad90-3ce782d831b3"/>
        <memory_policy>
            <overcommit percent="100"/>
            <transparent_hugepages>
                <enabled>true</enabled>
            </transparent_hugepages>
        </memory_policy>
        <scheduling_policy>
            <policy>none</policy>
        </scheduling_policy>
        <version major="3" minor="3"/>
        <error_handling>
            <on_error>migrate</on_error>
        </error_handling>
        <virt_service>false</virt_service>
        <gluster_service>true</gluster_service>
        <threads_as_cores>false</threads_as_cores>
        <tunnel_migration>false</tunnel_migration>
```

```
        <trusted_service>false</trusted_service>
        <ballooning_enabled>false</ballooning_enabled>
    </cluster>
</clusters>
```

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

Report a bug

3.3. Example: List Host Collection

This example retrieves a representation of the host collection and shows a Red Hat Storage server named **host1** registered with the default cluster.

Example 3.3. List hosts collection

```
GET /api/hosts HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* --cacert *[CERT]* https://*[RHSC Host]*/api/hosts

The API returns the following representation:

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

<hosts>
    <host href="/api/hosts/411fd862-1469-4dff-ad5a-a7be364d83a6"
id="411fd862-1469-4dff-ad5a-a7be364d83a6">
        <actions>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/approve" rel="approve"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/forceselectspm" rel="forceselectspm"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/iscsilogin" rel="iscsilogin"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/iscsidiscover" rel="iscsidiscover"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/commitnetconfig" rel="commitnetconfig"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/fence" rel="fence"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/deactivate" rel="deactivate"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/install" rel="install"/>
            <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/activate" rel="activate"/>
        </actions>
```

```
        <name>host1</name>
        <comment></comment>
        <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/nics" rel="nics"/>
        <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/tags" rel="tags"/>
        <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/permissions" rel="permissions"/>
        <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/statistics" rel="statistics"/>
        <link href="/api/hosts/411fd862-1469-4dff-ad5a-
a7be364d83a6/hooks" rel="hooks"/>
        <address>10.70.42.154</address>
        <certificate>
            <organization>TestOrg</organization>
            <subject>O=TestOrg,CN=10.70.42.154</subject>
        </certificate>
        <status>
            <state>up</state>
        </status>
        <cluster href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32" id="02b2bd03-5e54-45f9-9302-33a4ba96eb32"/>
        <port>54321</port>
        <type>rhel</type>
        <storage_manager priority="5">false</storage_manager>
        <version major="4" minor="14" build="7" revision="2"
full_version="vdsm-4.14.7.2-1.el6rhs"/>
        <hardware_information>
            <manufacturer>Red Hat</manufacturer>
            <version>6Server-6.4.0.4.el6</version>
            <serial_number>00000000-0000-0000-0000-
00259078DB32</serial_number>
            <product_name>RHEV Hypervisor</product_name>
            <uuid>504dd302-0c14-4153-ae48-a0b2fa9484ce</uuid>
            <family>Red Hat Enterprise Linux</family>
        </hardware_information>
        <power_management type="apc">
            <enabled>false</enabled>
            <options/>
        </power_management>
        <ksm>
            <enabled>false</enabled>
        </ksm>
        <transparent_hugepages>
            <enabled>true</enabled>
        </transparent_hugepages>
        <iscsi>
            <initiator>iqn.1994-05.com.redhat:6f8b848932c9</initiator>
        </iscsi>
        <ssh>
            <port>22</port>

<fingerprint>e7:02:ba:01:72:35:1c:5b:5a:ac:ba:b6:da:5d:23:8e</fingerpri
nt>
        </ssh>
        <cpu>
```

```
            <topology sockets="1" cores="1" threads="1"/>
            <name>Intel Xeon E312xx (Sandy Bridge)</name>
            <speed>2200</speed>
        </cpu>
        <memory>2103443456</memory>
<max_scheduling_memory>1698693120</max_scheduling_memory>
        <summary>
            <active>0</active>
            <migrating>0</migrating>
            <total>0</total>
        </summary>
        <os type="RHEL">
            <version full_version="6Server - 6.5.0.1.el6"/>
        </os>
        <libvirt_version major="0" minor="10" build="2" revision="0"
full_version="libvirt-0.10.2-29.el6_5.9"/>
    </host>
</hosts>
```

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

Report a bug

3.4. Example: Add Host to Cluster

This example adds a host to the default cluster and shows a host named *host1* registered with the storage environment.

Example 3.4. Add Server to Cluster

```
POST /api/hosts HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host>
  <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"/>
  <name>host1</name>
  <address>IP_ADDRESS</address>
  <root_password>ROOT_PASSWORD</root_password>
</host>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] --cacert [CERT] https://[RHSC
HOST]/api/hosts -d "<host><cluster id=\"99408929-82cf-4dc7-a532-
9d998063fa95\"/><name>host1</name><address>IP_ADDRESS</address>
<root_password>ROOT_PASSWORD</root_password></host>"
```

The API returns the following representation of the newly created server:

```
HTTP/1.1 201 Created
```

```
Content-Type: application/xml

<host href="/api/hosts/de173e6a-fb05-11e1-a2fc-0050568c4349"
id="de173e6a-fb05-11e1-a2fc-0050568c4349">
    <name>host1</name>
    <actions>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/fence" rel="fence"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/install" rel="install"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/activate" rel="activate"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/deactivate" rel="deactivate"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/approve" rel="approve"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/iscsilogin" rel="iscsilogin"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/iscsidiscover" rel="iscsidiscover"/>
        <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/commitnetconfig" rel="commitnetconfig"/>
    </actions>
    <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-0050568c4349/storage"
rel="storage"/>
    <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-0050568c4349/nics"
rel="nics"/>
    <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-0050568c4349/tags"
rel="tags"/>
    <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/permissions" rel="permissions"/>
    <link href="/api/hosts/de173e6a-fb05-11e1-a2fc-
0050568c4349/statistics" rel="statistics"/>
    <address>10.16.159.64</address>
    <status>
        <state>unassigned</state>
    </status>
    <cluster href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95"
id="99408929-82cf-4dc7-a532-9d998063fa95"/>
    <port>54321</port>
    <type>rhel</type>
    <storage_manager priority="5">false</storage_manager>
    <power_management>
        <enabled>false</enabled>
        <options/>
    </power_management>
    <ksm>
        <enabled>false</enabled>
    </ksm>
    <transparent_hugepages>
        <enabled>false</enabled>
    </transparent_hugepages>
    <cpu>
        <speed>0</speed>
    </cpu>
    <memory>0</memory>
```

```
    <summary>
        <total>0</total>
    </summary>
</host>
```

Report a bug

3.5. Example: Create Volume

This example creates a volume **data**, volume type **DISTRIBUTE**, and having two bricks in the default cluster.

Example 3.5. Creating a Volume

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

<gluster_volume>
  <name>data</name>
  <volume_type>DISTRIBUTE</volume_type>
  <bricks>
    <brick>
      <server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
      <brick_dir>/export/data/brick1</brick_dir>
    </brick>
    <brick>
      <server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
      <brick_dir>/export/data/brick2</brick_dir>
    </brick>
  </bricks>
</gluster_volume>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] --cacert [CERT] https://[RHSC
HOST]/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes -d
"<gluster_volume><name>data</name><volume_type>DISTRIBUTE</volume_type>
<bricks><brick><server_id>fcb46b88-f32e-11e1-918a-
0050568c4349</server_id><brick_dir>/export/data/brick1</brick_dir>
</brick><brick><server_id>de173e6a-fb05-11e1-a2fc-
0050568c4349</server_id><brick_dir>/export/data/brick2</brick_dir>
</brick></bricks></gluster_volume>"
```

The API returns the following representation of the newly created volume resource:

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

<gluster_volume href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-209cd1a8aafb"
```

```
id="6c657343-7a9d-46f1-b9f2-209cd1a8aafb">
    <actions>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-
209cd1a8aafb/setoption" rel="setoption"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-
209cd1a8aafb/resetoption" rel="resetoption"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-
209cd1a8aafb/resetalloptions" rel="resetalloptions"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-
209cd1a8aafb/rebalance" rel="rebalance"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-
209cd1a8aafb/stoprebalance" rel="stoprebalance"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-209cd1a8aafb/start"
rel="start"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-209cd1a8aafb/stop"
rel="stop"/>
    </actions>
    <name>vol1</name>
    <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/6c657343-7a9d-46f1-b9f2-209cd1a8aafb/bricks"
rel="bricks"/>
    <cluster href="/api/clusters/02b2bd03-5e54-45f9-9302-33a4ba96eb32"
id="02b2bd03-5e54-45f9-9302-33a4ba96eb32"/>
    <volume_type>distribute</volume_type>
    <transport_types>
        <transport_type>tcp</transport_type>
    </transport_types>
    <replica_count>0</replica_count>
    <stripe_count>0</stripe_count>
    <options/>
    <status>
        <state>down</state>
    </status>
</gluster_volume>
```

Report a bug

3.6. Example: List Volume Collection

This example retrieves a representation of the volume collection and shows a list of volumes present in the **`Default`** cluster.

Example 3.6. List Volume Collection

```
GET /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes
HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* https://*[RHSC Host]*/api/clusters/*99408929-82cf-4dc7-a532-9d998063fa95*/glustervolumes

The API returns the following representation:

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

<gluster_volumes>
    <gluster_volume href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa"
id="0635fb7e-0da8-48ca-ae9c-72be85c36afa">
        <actions>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/setoption" rel="setoption"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/resetoption" rel="resetoption"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/resetalloptions" rel="resetalloptions"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/rebalance" rel="rebalance"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/stoprebalance" rel="stoprebalance"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa/start"
rel="start"/>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa/stop"
rel="stop"/>
        </actions>
        <name>test</name>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks" rel="bricks"/>
        <cluster href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32" id="02b2bd03-5e54-45f9-9302-33a4ba96eb32"/>
        <volume_type>distribute</volume_type>
        <transport_types>
            <transport_type>tcp</transport_type>
        </transport_types>
        <replica_count>0</replica_count>
        <stripe_count>0</stripe_count>
        <options>
            <option name="auth.allow" value="*"/>
            <option name="nfs.disable" value="off"/>
```

```
            <option name="user.cifs" value="enable"/>
        </options>
        <status>
            <state>up</state>
        </status>
    </gluster_volume>
</gluster_volumes>
```

Report a bug

3.7. Example: Start Volume

This example retrieves a representation of starting a volume.

Example 3.7. Start Volume

```
POST api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/start
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] https://[RHSC
HOST]/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/start -d
"<action/>"
```

The API returns the following representation:

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

<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

3.8. Example: List Brick Collection

This example retrieves a representation of the brick collection and shows a list of bricks of a volume.

Example 3.8. List Brick Collection

```
GET /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks
HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* https://*[RHSC HOST]*/api/clusters/*99408929-82cf-4dc7-a532-9d998063fa95*/glustervolumes/*83101900-2f12-4855-838e-36b8a9e04554*/bricks

The API returns the following representation:

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

<bricks>
    <actions>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/migrate" rel="migrate"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/stopmigrate" rel="stopmigrate"/>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/activate" rel="activate"/>
    </actions>
    <brick href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/f115ae70-2ff6-43c7-aa36-26631df82bcb" id="f115ae70-
2ff6-43c7-aa36-26631df82bcb">
        <actions>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/f115ae70-2ff6-43c7-aa36-26631df82bcb/replace"
rel="replace"/>
        </actions>
        <name>vm12.lab.eng.blr.redhat.com:/home/1</name>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/f115ae70-2ff6-43c7-aa36-26631df82bcb/statistics"
rel="statistics"/>
        <gluster_volume href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa"
id="0635fb7e-0da8-48ca-ae9c-72be85c36afa"/>
        <server_id>411fd862-1469-4dff-ad5a-a7be364d83a6</server_id>
        <brick_dir>/home/1</brick_dir>
        <status>
            <state>up</state>
        </status>
    </brick>
    <brick href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/b46215cd-785e-41aa-b424-5783955d43ee" id="b46215cd-
785e-41aa-b424-5783955d43ee">
```

```
        <actions>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/b46215cd-785e-41aa-b424-5783955d43ee/replace"
rel="replace"/>
        </actions>
        <name>vm12.lab.eng.blr.redhat.com:/home/2</name>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/b46215cd-785e-41aa-b424-5783955d43ee/statistics"
rel="statistics"/>
        <gluster_volume href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa"
id="0635fb7e-0da8-48ca-ae9c-72be85c36afa"/>
        <server_id>411fd862-1469-4dff-ad5a-a7be364d83a6</server_id>
        <brick_dir>/home/2</brick_dir>
        <status>
            <state>up</state>
        </status>
    </brick>
</bricks>
```

Report a bug

3.9. Example: Add Bricks to Volume

This example adds a brick to volume.

Example 3.9. Add Bricks to Volume

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<bricks>
  <brick>
<server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
    <brick_dir>/export/data/brick3</brick_dir>
  </brick>
</bricks>
```

cURL command:

```
 curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC
HOST]/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks -
d "<bricks><brick><server_id>fcb46b88-f32e-11e1-918a-
0050568c4349</server_id><brick_dir>/export/data/brick3</brick_dir>
</brick></bricks>"
```

The API returns the following representation:

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

<bricks>
    <brick href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/5241646b-f7aa-4484-9c4a-be33ebf4f51d" id="5241646b-
f7aa-4484-9c4a-be33ebf4f51d">
        <actions>
            <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/5241646b-f7aa-4484-9c4a-be33ebf4f51d/replace"
rel="replace"/>
        </actions>
        <name>vm12.lab.eng.blr.redhat.com:/home/3</name>
        <link href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-
72be85c36afa/bricks/5241646b-f7aa-4484-9c4a-be33ebf4f51d/statistics"
rel="statistics"/>
        <port>49154</port>
        <pid>26691</pid>
        <device>/dev/mapper/vg_vm12-lv_root</device>
        <mnt_options>rw</mnt_options>
        <fs_name>ext4</fs_name>
        <gluster_clients>
            <gluster_client>
                <host_name>10.70.36.84</host_name>
                <client_port>1019</client_port>
                <bytes_read>1260</bytes_read>
                <bytes_written>900</bytes_written>
            </gluster_client>
        </gluster_clients>
        <memory_pools>
            <memory_pool>
                <name>test-server:fd_t</name>
                <hot_count>0</hot_count>
                <cold_count>1024</cold_count>
                <padded_size>108</padded_size>
                <alloc_count>0</alloc_count>
                <max_alloc>0</max_alloc>
                <pool_misses>0</pool_misses>
                <max_stdalloc>0</max_stdalloc>
            </memory_pool>
            ...
        </memory_pools>
        <gluster_volume href="/api/clusters/02b2bd03-5e54-45f9-9302-
33a4ba96eb32/glustervolumes/0635fb7e-0da8-48ca-ae9c-72be85c36afa"
id="0635fb7e-0da8-48ca-ae9c-72be85c36afa"/>
        <server_id>411fd862-1469-4dff-ad5a-a7be364d83a6</server_id>
        <brick_dir>/home/3</brick_dir>
        <status>
```

```
            <state>up</state>
        </status>
    </brick>
</bricks>
```

Report a bug

3.10. Example: Check System Events

The **login** action for **admin** creates entries in the **events** collection. This example lists the events collection and identifies events specific to log in of the admin.

Example 3.10. List the events collection

```
GET /api/events HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* --cacert *[CERT]* https://*[RHSC Host]*/api/events

The API returns a representation that includes the following:

```
<events>
    <event href="/api/events/54" id="54">
        <description>User admin@internal logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2013-11-27T17:48:01.264+05:30</time>
        <user href="/api/users/fdfc627c-d875-11e0-90f0-83df133b58cc"
id="fdfc627c-d875-11e0-90f0-83df133b58cc"/>
        <origin>oVirt</origin>
        <custom_id>-1</custom_id>
        <flood_rate>30</flood_rate>
    </event>
</events>
```

The following event occurs:

- **id="54"** - The API authenticates with the **admin** user's username and password.
- **id="192"** - The API, acting as the **admin** user, starts **Volume Data** on the **Default** cluster
- **id="193"** - The API logs out of the **admin** user account.

Report a bug

Chapter 4. Python Quick Start Example

4.1. Python Quick Start Introduction

This chapter provides a series of examples demonstrating the steps to create a cluster within a basic Red Hat Storage environment using the Python Software Development Kit.

These examples use the **ovirtsdk** Python library provided by the **rhsc-sdk** package. This package is available to systems subscribed to a Red Hat Storage pool if you use the certificate-based Red Hat Network, or the Red Hat Storage channel if you use the Red Hat Network classic. See the *Red Hat Storage Console Installation Guide* for more information on subscribing systems to download software from these locations.

Note

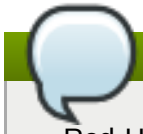
See the *Red Hat Storage Console Installation Guide* for specific channel names current to your system.

You will also need:

- A networked installation of Red Hat Storage Console.
- A networked and configured Red Hat Storage Server.
- A working understanding of both the logical and physical objects that make up a Red Hat Storage environment.
- A working understanding of the Python programming language.

Important

All Python examples include placeholders for authentication details (USER for user name, and PASS for password). Ensure all requests performed with Python fulfill the authentication requirements of your environment.



Note

Red Hat Storage generates a globally unique identifier (GUID) for the id attribute for each resource. Identifier codes in these examples might appear different to the identifier codes in your Red Hat Storage environment.

Note

These Python examples contain only basic exception and error handling logic. For more information on the exception handling specific to the SDK, see the pydoc for the **ovirtsdk.infrastructure.errors** module.

Report a bug

4.2. Example: Accessing the API Entry Point using Python

The **ovirtsdk** Python library provides the API class, which acts as the entry point for the API.

Example 4.1. Accessing the API entry point using Python

To connect the example creates an instance of the **API** class. If connection is established successfully, a message is printed. Lastly, the **disconnect()** method of the **API** class is called to close the connection.

The parameters provided to the constructor for the API class in this example are:

- The **url** of the Console with which to connect.
- The **username** of the user to authenticate.
- The **password** of the user to authenticate.
- The **ca_file** that is the path to a certificate. The certificate is expected to be a copy of the Console's Certificate Authority. It can be obtained from **https://*[HOST]*/ca.crt**.

The constructor for the **API** class supports other parameters. Only mandatory parameters are specified in this example.

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
               username="USER",
               password="PASS",
               ca_file="ca.crt")

    print "Connected to %s successfully!" % api.get_product_info().name

    api.disconnect()

except ConnectionError, err:
    print "Connection failed: %s" % err
```

If the connection attempt was successful, the example outputs the text:

```
Connected to Red Hat Storage Console successfully!
```

Example 4.2. Listing the Cluster Collection using Python

The **API** class provides a cluster collection named default cluster. This collection contains all the clusters in the environment.

This Python example lists the clusters in the default cluster collection.

```
from ovirtsdk.api import API
```

```
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
               username="USER",
               password="PASS",
               ca_file="ca.crt")

    c_list = api.clusters.list()

    for c in c_list:
        print "%s (%s)" % (c.get_name(), c.get_id())

    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

In an environment where only the **Default** cluster exists, the example outputs:

```
Default (99408929-82cf-4dc7-a532-9d998063fa95)
```

Example 4.3. Listing the Networks Collection using Python

The **API** class provides access to a networks collection named **Management Networks**. This collection contains all the networks in the environment.

This Python example lists the networks in the **networks** collection. It also outputs some basic information about each network in the collection.

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER@domain",
          password="PASS",
          ca_file="ca.crt")

    n_list = api.networks.list()

    for n in n_list:
        print n.get_description()
        print n.get_id()
        print n.get_name()

    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

In an environment where only the default management network exists, this example outputs the description of the network, the network ID and the name of the network:

```
Management Network
00000000-0000-0000-0000-000000000009
ovirtmgmt
```

Example 4.4. Listing the Host Collection using Python

The **API** class provides access to a host collection. This collection contains all the hosts in the storage cluster.

This Python example lists the hosts in the host collection.

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER@domain",
          password="PASS",
          ca_file="ca.crt")

    s_list = api.hosts.list()

    for s in s_list:
        print "host name: %s (host ID: %s)" % (s.get_name(),
s.get_id())

    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

The code snippet displays the list of hosts in the environment along with the associated host IDs.

```
host name: 10.70.37.49 (host ID: 5be18d62-e7b0-4407-8ff6-68290a92338f)
host name: 10.70.37.50 (host ID: 3b202041-6e14-43df-a844-92a141bed1ed)
```

Example 4.5. Listing the Volume Collection using Python

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER@domain",
          password="PASS",
          ca_file="ca.crt")

    clusterName= "CLUSTERNAME"
    for volume in api.clusters.get(
        clusterName).glustervolumes.list():
        print "volumeName:", volume.get_name(),
```

```
    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

Example 4.6. Listing the Brick Collection from a Volume using Python

The function **`getGlusterVolumeBrickDetails()`** lists the bricks of the volume that is assigned to the cluster.

```
def getGlusterVolumeBrickDetails(clusterName, volumeName):
    bricks = []
    for brick in API.clusters.get(
        clusterName).glustervolumes.get(
            volumeName).get_bricks().list():
        bricks.append({"Brick": brick.get_name(),
                       "Status": brick.get_status().state})
    return bricks
```

Example 4.7. Listing hooks in a Red Hat Storage Console

You can view the volume lifecycle extensions in a Red Hat Storage Console using Gluster hooks. The function **`getHookList(clusterName)`** lists the Gluster hooks created in the Console.

```
from ovirtsdk.api import API
from ovirtsdk.xml import params
try:
    api = API (url="https://HOST",
          username="USER@domain",
          password="PASS",
          ca_file="ca.crt")

    clusterName="Default"
    hooks=[]
    for hook in api.clusters.get(clusterName).glusterhooks.list():
        print "hookName:", hook.name
        print "glusterCommand:", hook.get_gluster_command()
        print "Stage:", hook.get_stage()
        print "Hook ID:", hook.get_id()

    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

The code snippet displays the list of hook scripts in the console along with the associated hook IDs,

```
hookName: add-brick-PRE-28Quota-enable-root-xattr-heal.sh
glusterCommand: add-brick
Stage: PRE
Hook ID: b368f9dc-02a5-4e86-b96e-ea5393e73dc7
```

```
hookName: gsync-create-POST-56glusterd-geo-rep-create-post.sh
glusterCommand: gsync-create
Stage: POST
```

Report a bug

Part II. REST Application Programming Interface

Chapter 5. 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 5.1. Accessing the API Entry Point

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

```
GET /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 href="/api/domains" rel="domains" />
     ...
        <product_info>
        <name>Red Hat Storage Console</name>
        <vendor>Red Hat</vendor>
        <version major="3" minor="0" build="0" revision="0"/>
        <full_version>3.0.0-0.10.el6_5</full_version>
    </product_info>
    <summary>
        <hosts>
            <total>3</total>
            <active>2</active>
        </hosts>
        <users>
            <total>1</total>
            <active>1</active>
        </users>
    </summary>
</api>
```

Note

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

Report a bug

5.1. Product Information

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

Example 5.2. Verify a genuine Red Hat Storage environment

The follow elements identify a genuine Red Hat Storage 3.0 environment:

```
<api>
    ...
      <product_info>
        <name>Red Hat Storage Console</name>
        <vendor>Red Hat</vendor>
        <version major="3" minor="0" build="0" revision="0"/>
        <full_version>3.0.0-0.10.el6_5</full_version>
    </product_info>

    ...
</api>
```

Report a bug

5.2. 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 5.1. Available Relationship Types

Relationship	Description
capabilities	Supported capabilities of the Red Hat Storage Console.
clusters	Clusters.
events	Events.
hosts	Hosts.
networks	Virtual networks.
roles	Roles.
tags	Tags.
users	Users.
groups	Imported identity service groups.
domains	Identity service domains.

The **link** elements also contain a set of **search** URIs for certain collections. These URIs use URI templates [2] 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 5.2. Relationships associated with search query URIs

Relationship	Description
clusters/search	Query clusters.
events/search	Query events.
hosts/search	Query hosts.
users/search	Query users.
groups/search	Query groups.

Report a bug

5.3. Summary element

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

Table 5.3. Summary Elements

Element	Description
hosts	Total number of hosts and total number of active hosts.
users	Total number of users and total number of active users.

Report a bug

5.4. RESTful Service Description Language (RSDL)

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

```
GET /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="/api?rsdl" rel="rsdl">
    <description>...</description>
    <version major="3" minor="0" build="0" revision="0"/>
    <schema href="/api?schema" rel="schema">
        <name>...</name>
        <description>...</description>
    </schema>
    <links>
        <link href="/api/capabilities" rel="get">
            ...
        </link>
        ...
    </links>
</rsdl>
```

Table 5.4. Table 5.5. RSDL Structure Elements

Element	Description
description	A plain text description of the RSDL document.

Element	Description
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="/api?rsdl" rel="rsdl">
    ...
    <links>
        <link href="/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 5.5. 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.

Element	Description
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 Storage Console 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 Storage environment.

Report a bug

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

Chapter 6. Capabilities

The capabilities collection provides information about the supported capabilities of a Red Hat Storage Console version. These capabilities include active features and available enumerated values for specific properties. An API user accesses this information through the **rel="capabilities"** link obtained from the entry point URI.

Report a bug

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 Storage Console 3.0 and 2.1 respectively:

```
<capabilities>
    <version major="3" minor="2">
        ...
    </version>
    <version major="3" minor="1">
        ...
    </version>
    ...
</capabilities>
```

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

Report a bug

6.2. Current Version

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

```
<capabilities>
    <version major="3" minor="4" href="/api/capabilities/332e3433-2e34-
332e-3433-2e34332e3433" id="332e3433-2e34-332e-3433-2e34332e3433">
        ...
        <current>true</current>
        ...
    </version>
</capabilities>
```

Report a bug

6.3. Red Hat Storage Volume Types

The **gluster_volume_types** element lists the available type of Red Hat Storage volumes.

```
<capabilities>
    <version major="3" minor="4"href="/api/capabilities/332e3433-2e34-
332e-3433-2e34332e3433" id="332e3433-2e34-332e-3433-2e34332e3433">
        ...
        <gluster_volume_types>
            <gluster_volume_type>distribute</gluster_volume_type>
<gluster_volume_type>replicate</gluster_volume_type>
<gluster_volume_type>distributed_replicate</gluster_volume_type>
<gluster_volume_type>stripe</gluster_volume_type>
<gluster_volume_type>distributed_stripe</gluster_volume_type>
<gluster_volume_type>striped_replicate</gluster_volume_type>
<gluster_volume_type>distributed_striped_replicate</gluster_volume_type>
        </gluster_volume_types>
        ...
    </version>
</capabilities>
```

Report a bug

6.4. Red Hat Storage Transport Types

The **transport_types** element lists the available transport types for Red Hat Storage volumes.

```
<capabilities>
    <version major="3" minor="4" href="/api/capabilities/332e3433-2e34-
332e-3433-2e34332e3433" id="332e3433-2e34-332e-3433-2e34332e3433">
        ...
        <transport_types>
            <transport_type>tcp</transport_type>
            <transport_type>rdma</transport_type>
        </transport_types>
        ...
    </version>
</capabilities>
```

Report a bug

6.5. Step Type

The **step_type** element lists the available type of job steps while monitoring a jobs on Red Hat Storage volumes.

```
<step_types>
            <step_type>validating</step_type>
            <step_type>executing</step_type>
            <step_type>finalizing</step_type>
            <step_type>rebalancing_volume</step_type>
            <step_type>removing_bricks</step_type>
            <step_type>unknown</step_type>
</step_types>
```

Report a bug

6.6. Gluster Hook Content Type

The **content_type** element lists the Gluster Hook content types.

```
      <content_types>
            <content_type>text</content_type>
            <content_type>binary</content_type>
        </content_types>
```

Report a bug

6.7. Gluster Hook Stage

The **hook_states** element lists the available status of the hooks.

```
      <hook_states>
            <hook_state>enabled</hook_state>
            <hook_state>disabled</hook_state>
            <hook_state>missing</hook_state>
        </hook_states>
```

Report a bug

6.8. Resource Status States

Each version contains a set of states for resource statuses. These resource status elements include **creation_states**, **host_states**, **host_non_operational_details**, **gluster_volume_states**, and **brick_states**.

Report a bug

Chapter 7. Common Features

This chapter examines features common to resources and collections.

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-updateable	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.	

Report a bug

7.1. Representations

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

```
<resource id="resource_id" href="/api/collection/resource_id">
    <name>Resource-Name</name>
    ...
</resource>
```

In the context of a volume, the representation appears as follows:

```
<gluster_volume href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554"
id="83101900-2f12-4855-838e-36b8a9e04554">
        <name>vol1</name>
        ...
</gluster_volume>
```

All resource representations contain a set of common attributes

Table 7.2. Common attributes to resource representations

Attribute	Type	Description	Properties

Attribute	Type	Description	Properties
id	GUID	Each resource in the storage 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.	

Report a bug

7.2. Collections

A collection is a set of resources of the same type. The API provides both top-level collections and sub-collections. This section examines common features for collections.

Report a bug

7.2.1. Listing All Resources in a Collection

A listing of the resources in a collection is obtained by issuing a **GET** request on the collection URI obtained from the entry point.

```
GET /api/[collection] HTTP/1.1
Accept: application/xml

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

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

Report a bug

7.2.2. Listing Extended Resource Sub-Collections

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

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

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

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

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

Table 7.3. Collections that use extended sub-collections

Collection	Extended Sub-Collection Support
hosts	**statistics**
bricks	**statistics**
step	**statistics**

Example 7.1. An request for extended statistics in the servers collection

```
GET /api/hosts HTTP/1.1
Accept: application/xml; detail=statistics
```

Report a bug

7.2.3. 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 /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="/api/collection/resource_id">
        ...
    </resource>
    ...
</collection>
```

Report a bug

7.2.3.1. Query Syntax

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

```
GET /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 Storage Console search query language:

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

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

Table 7.4. Example search queries

Collection	Criteria	Result
volumes	**type=REPLICATE**	Displays a list of all replicate volumes
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 7.2. URL-encoded search query

```
GET /api/events?search=severity%3Derror HTTP/1.1
Accept: application/xml
```

Important

All search queries are case-sensitive.

Report a bug

7.2.3.2. Wildcards

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

Example 7.3. Wildcard search query for name=server*

```
GET /api/hosts?search=name%3Dserver* HTTP/1.1
Accept: application/xml
```

This query would result in all servers with names beginning with **server**, such as **server-1**, **server-2**, or **server-data**.

Example 7.4. Wildcard search query for name=s*1

```
GET /api/hosts?search=name%3D*1 HTTP/1.1
Accept: application/xml
```

> This query would result in all servers with names beginning with **s** and ending with **1**, such as **server1** or **server-1** .

Report a bug

7.2.3.3. Pagination

Some Red Hat Storage 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 7.5. Paginating resources**
>
> This example paginates resources in a collection. The URL-encoded request is:
>
> ```
> GET /api/collection?search=page%201 HTTP/1.1
> Accept: application/xml
> ```
>
> Increase the **page** value to view the next page of results.
>
> ```
> GET /api/collection?search=page%202 HTTP/1.1
> Accept: application/xml
> ```
>
> Use the **page** command also in conjunction with other commands in a search query. For example:
>
> ```
> GET /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.

Report a bug

7.2.4. Creating a Resource in a Collection

The API creates 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: application/xml** header. This informs the API of the XML representation in the body content as part of the request.

Each resource type has its own specific required properties. The client supplies these properties as XML elements 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 **fault** representation indicating the missing elements.

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

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

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

<resource id="resource_id" href="/api/collection/resource_id">
    <name>Resource-Name</name>
    ...
</resource>
```

The **`Location`** header in the response gives the URI of the queried resource. The response body contains either a complete representation, partial representation or no representation of the resource. It is recommended that clients rely only on fetching the representation via the URI in the response header.

Report a bug

7.3. Resources

This section examines common features for resources.

Report a bug

7.3.1. Retrieving a Resource

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

```
GET /api/collection/resource_id HTTP/1.1
Accept: application/xml

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

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

Report a bug

7.3.2. Updating a Resource

The API modifies 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: application/xml`** header. This informs the API of the XML representation in the body content as part of the request.

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

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

```
</resource>

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

<resource id="resource_id" href="/api/collection/resource_id">
    <name>New-Resource-Name</name>
    ...
</resource>
```

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 **`409 Conflict`** error with a **`fault`** representation in the response body.

Properties omitted from the representation are ignored and not changed.

Report a bug

7.3.3. Deleting a Resource

The API deletes a resource with a **`DELETE`** request sent to its URI.

```
DELETE /api/collection/resource_id HTTP/1.1
Accept: application/xml

HTTP/1.1 204 No Content
```

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: application/xml`** header to inform the API of the XML representation in the body content. If a **`DELETE`** request contains no body content, omit the **`Content-Type: application/xml`** header.

Report a bug

7.3.4. 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 volume contains bricks, which means the API maps the relationship between the volume resource and the bricks sub-collection.

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

- 1:N mappings, where mapped resources are dependent on a parent resources. Without the parent resource, the dependent resource cannot exist. For example, the link between a volume and its bricks.
- 1:N mappings, where mapped resources exist independently from parent resources but data is still associated with the relationship. For example, the link between a network and a cluster.

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

```
GET /api/collection/resource_id HTTP/1.1
Accept: application/xml
```

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

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

The API user now queries the sub-collection.

```
GET /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="/api/collection/resource_id/subcollection/subresource_id">
        ...
    </subresource>
    ...
</subcollection>
```

Report a bug

7.3.5. 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 7.6. Backlinking from a sub-collection resource to a resource using an XML element

```
GET /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="/api/collection/resource_id/subcollection/subresource_id">
```

```
        <resource id="resource_id" href="/api/collection/resource_id"/>
        ...
    </subresource>
</subcollection>
```

Report a bug

7.3.6. 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="/api/collection/resource_id/start"/>
        <link rel="stop" href="/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 7.5. Common action parameters

Element	Description
async	If **true**, 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 /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="/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 7.6. 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 /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="/api/collection/resource_id/action/action_id">
    <status>
        <state>pending</state>
    </status>
    <link rel="parent" /api/collection/resource_id"/>
    <link rel="replay" href="/api/collection/resource_id/action"/>
<action>
```

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

Table 7.7. Action relationships

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

Report a bug

7.3.7. Permissions

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

```
GET /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="/api/collection/resource_id/permissions/permission_id">
        <role id="role_id" href="/api/roles/role_id"/>
        <user id="user_id" href="/api/users/user_id"/>
        <resource id="resource_id" href="/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 /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="/api/resources/resource_id/permissions/permission_id">
    <role id="role_id" href="/api/roles/role_id"/>
    <user id="user_id" href="/api/users/user_id"/>
    <resource id="resource_id" href="/api/collection/resource_id"/>
</permission>
```

Report a bug

7.3.8. 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 /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>
```

Report a bug

Chapter 8. Clusters

The **clusters** collection provides information about clusters in a Red Hat Storage environment. An API user accesses this information through the **rel="clusters"** link obtained from the entry point URI (see Chapter 5, *Entry Point*).

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

Table 8.1. Cluster elements

Element	Type	Description	Properties
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. See Section 7.3.7, “Permissions”.	
version major= minor=	complex	The compatibility level of the cluster.	
supported_versions	complex	A list of possible **version** levels for the cluster.	
gluster_service	boolean	defines whether gluster services are enabled on the cluster. Is always true in Red Hat Storage Console	
virt_service	boolean	defines whether virtualization services are enabled on the cluster. Is always false in Red Hat Storage Console.	

Example 8.1. An XML representation of a cluster

```
  <clusters>
    <cluster href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95"
id="99408929-82cf-4dc7-a532-9d998063fa95">
        <name>Default</name>
        <description>The default server cluster</description>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/networks" rel="networks"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/permissions" rel="permissions"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes" rel="glustervolumes"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glusterhooks" rel="glusterhooks"/>
        <data_center href="/api/datacenters/5849b030-626e-47cb-ad90-
3ce782d831b3" id="5849b030-626e-47cb-ad90-3ce782d831b3"/>
        <memory_policy>
            <overcommit percent="100"/>
            <transparent_hugepages>
                <enabled>true</enabled>
            </transparent_hugepages>
        </memory_policy>
```

```
        <scheduling_policy>
            <policy>none</policy>
        </scheduling_policy>
        <version major="3" minor="4"/>
        <error_handling>
            <on_error>migrate</on_error>
        </error_handling>
        <virt_service>false</virt_service>
        <gluster_service>true</gluster_service>
        <threads_as_cores>false</threads_as_cores>
        <tunnel_migration>false</tunnel_migration>
        <trusted_service>false</trusted_service>
        <ballooning_enabled>false</ballooning_enabled>
        <ksm>
            <enabled>true</enabled>
        </ksm>
    </clusters>
```

Creation of a new cluster requires the **name** and **server** elements. Identify the **server** with **id** attribute and **name** element. **name** element is mandatory. See Section 7.2.4, “Creating a Resource in a Collection” for more information.

Example 8.2. Creating a cluster

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

<cluster>
    <name>cluster1</name>
 </cluster>
```

Example 8.3. Updating a cluster

```
PUT /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95 HTTP/1.1
Accept: application/xml
Content-type: application/xml

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

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

Example 8.4. Removing a cluster

```
DELETE /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95 HTTP/1.1

HTTP/1.1 204 No Content
```

Report a bug

8.1. 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 with an additional **cluster id=** to signify a relationship to the cluster and a **display** element to represent the display network status in the cluster.

An API user manipulates the **networks** sub-collection as described in Chapter 7, *Common Features*. **POST**ing a network **id** or **name** reference to the **networks** sub-collection associates the network with the cluster.

Example 8.5. Associating a network resource with a cluster

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

<network>
    <name>ovirtmgmt</name>
</network>

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

<network id="da05ac09-00be-45a1-b0b5-4a6a2438665f"
  href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/networks/
  da05ac09-00be-45a1-b0b5-4a6a2438665f">
    <name>rhsc</name>
    <status>
        <state>operational</state>
    </status>
    <description>Display Network</description>
    <cluster id="99408929-82cf-4dc7-a532-9d998063fa95"
      href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95"/>
    <data_center id="d70d5e2d-b8ad-494a-a4d2-c7a5631073c4"
      href="/api/datacenters/d70d5e2d-b8ad-494a-a4d2-c7a5631073c4"/>
    <display>true</display>
</network>
```

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

Example 8.6. Setting the display network status

```
PUT /api/clusters/99408929-82cf-4dc7-a532-
```

```
9d998063fa95/networks/da05ac09-00be-45a1-b0b5-4a6a2438665f HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<network>
    <display>false</display>
</network>
```

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

Example 8.7. Removing a network association from a cluster

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

Report a bug

8.2. Gluster Hooks

The **glusterhooks** collection provides information about Gluster Hooks in a Red Hat Storage environment. An API user accesses this information through the **rel="glusterhooks"** link obtained from the entry point URI (see Chapter 5, *Entry Point*).

Note

Gluster Hooks data from the storage cluster is collected periodically every two hours.

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

Table 8.2. Gluster Hooks elements

Element	Type	Description	Properties
cluster	relationship	A reference to the cluster that includes this hook.	locked
name	string	The name of the hook script	Read only
gluster_command	string	gluster volume command for which the hook is executed	read only
stage	string	stage of the command; **PRE** or **POST**	read only
content_type	string	**TEXT** or **BINARY**	read only
checksum	string	md5 checksum of the hook script	read only
content	string	content of the hook script	read only

Element	Type	Description	Properties
conflict_status	integer	bit representation of the conflict detected for the hook. First bit is for content, second is for status and third is for missing hooks.	read only
conflicts	string	comma separated list of conflict	read only
status	One of enabled or disabled	The status of the hook.	update

Example 8.8. An XML representation of Gluster Hook Collection

```
<glusterhooks>
    <gluster_hook href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6"
id="8ead05b0-3085-41a3-8693-9a7dfd6761a6">
        <actions>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/resolve"
rel="resolve"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/enable"
rel="enable"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/disable"
rel="disable"/>
        </actions>        <name>add-brick-PRE-28Quota-enable-root-
xattr-heal.sh</name>        <cluster href="/api/clusters/419590b8-5aa0-
473b-9651-aa41f1372c59" id="419590b8-5aa0-473b-9651-aa41f1372c59">
            <name>Cluster_3_3</name> ...
        </cluster>
        <gluster_command>add-brick</gluster_command>
        <stage>PRE</stage>
        <content_type>TEXT</content_type>
<checksum>f08a82c0ac88b2565842d1de7cdb14c0</checksum>
        <conflict_status>0</conflict_status>
        <conflicts></conflicts>
        <status>
            <state>enabled</state>
        </status>
    </gluster_hook>
```

Report a bug

8.2.1. Managing Gluster Hooks

This section describes the Gluster Hooks operations of the Red Hat Storage Console API such as listing, enabling, resolving, and removing them from a cluster.

Report a bug

8.2.1.1. Listing Gluster Hooks

A listing of hooks in the cluster along with conflict status is obtained by issuing a **GET** request on the volume **URI**.

```
GET /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glusterhooks
HTTP/1.1
Accept: application/xml
```

The API returns the following representation:

```
HTTP/1.1 200 OK
Accept: application/xml
<glusterhooks>
    <gluster_hook href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6"
id="8ead05b0-3085-41a3-8693-9a7dfd6761a6">
        <actions>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/resolve"
rel="resolve"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/enable"
rel="enable"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/8ead05b0-3085-41a3-8693-9a7dfd6761a6/disable"
rel="disable"/>
        </actions>
        <name>add-brick-PRE-28Quota-enable-root-xattr-heal.sh</name>
        <cluster href="/api/clusters/419590b8-5aa0-473b-9651-aa41f1372c59"
id="419590b8-5aa0-473b-9651-aa41f1372c59">
            <name>Cluster_3_3</name>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/networks" rel="networks"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/permissions" rel="permissions"/>

            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes" rel="glustervolumes"/>
            <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks" rel="glusterhooks"/>
            <memory_policy>
                <overcommit percent="100"/>
                <transparent_hugepages>
                    <enabled>true</enabled>
                </transparent_hugepages>
            </memory_policy>
            <scheduling_policy>
                <policy>none</policy>
            </scheduling_policy>
            <error_handling>
                <on_error>migrate</on_error>
            </error_handling>
            <virt_service>false</virt_service>
            <gluster_service>true</gluster_service>
            <threads_as_cores>false</threads_as_cores>
```

```
            <tunnel_migration>false</tunnel_migration>
            <trusted_service>false</trusted_service>
            <ballooning_enabled>false</ballooning_enabled>
 </cluster>
        <gluster_command>add-brick</gluster_command>
        <stage>PRE</stage>
        <content_type>TEXT</content_type>
        <checksum>f08a82c0ac88b2565842d1de7cdb14c0</checksum>
        <conflict_status>0</conflict_status>
        <conflicts></conflicts>
        <status>
            <state>enabled</state>
        </status>
    </gluster_hook>
```

Report a bug

8.2.1.2. Enabling Gluster Hooks

A Gluster Hook is enabled by issuing a **POST** request to its **URI**.

```
POST /api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glusterhooks/747bbb9e-ace6-424b-be31-16b33e02d882/enable
HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X POST -H "Accept: application/xml" -H "Content-Type: application/xml" -u *[USER:PASS]* https://*[RHSC HOST]*/api/clusters/419590b8-5aa0-473b-9651-aa41f1372c59/glusterhooks/747bbb9e-ace6-424b-be31-16b33e02d882/enable -d "<action/>"

The API returns the following representation:

```
<action>
    <job href="/api/jobs/58584575-6089-4cef-bdae-a2fc5f406bbd"
id="58584575-6089-4cef-bdae-a2fc5f406bbd"/>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

8.2.1.3. Resolving Gluster Hook Conflict

Gluster Hooks conflict is resolved by issuing a **POST** request. **MISSING_HOOK**, **CONTENT_CONFLICT**, and **STATUS_CONFLICT** are the types of conflict that can be resolved on a cluster.

- **Missing Hook Conflict**:

  MISSING_HOOK conflict can be resolved either by adding the hook to the host where it is missing or by removing it from all hosts and from the Red Hat Storage Console by issuing a **DELETE** request.

cURL command to resolve missing hook conflict by adding hook:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC
HOST]/api/clusters/44571c63-a110-4fba-9a8c-
7b30446ba8bf/glusterhooks/59ecbacb-1ce3-43f7-82e4-55db8ed74f56/resolve
-d "<action><resolution_type>ADD</resolution_type></action>"
```

The API returns the following representation:

```
<action>
    <resolution_type>ADD</resolution_type>
    <job href="/api/jobs/46d89857-37a1-492a-9327-78922884a778"
id="46d89857-37a1-492a-9327-78922884a778"/>
    <status>
        <state>complete</state>
    </status>
</action>
```

cURL command to resolve missing hook conflict by deleting hook:

```
curl -X DELETE -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC
HOST]/api/clusters/44571c63-a110-4fba-9a8c-
7b30446ba8bf/glusterhooks/18782869-3e62-42a5-a59d-91f896afd5d7
```

```
<action>
  <job href="/api/jobs/3dd9800f-a8c6-4f58-a3cf-dff190d2f06f"
id="3dd9800f-a8c6-4f58-a3cf-dff190d2f06f"/>
  <status><state>complete</state>
   </status>
</action>
```

- **Content Conflict**

This conflict can be resolved with the **resolution_type = copy**. If the version of the hook is incorrect in Red Hat Storage Console, the content is copied from a host by passing the *host.name* parameter. If the *host.name* parameter is empty, then the content is copied from Red Hat Storage Console to all host where content is different

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC
HOST]/api/clusters/44571c63-a110-4fba-9a8c-
7b30446ba8bf/glusterhooks/59ecbacb-1ce3-43f7-82e4-55db8ed74f56/resolve
-d "<action><resolution_type>COPY</resolution_type><host>
<name>host.name</name></host></action>"
```

The API returns the following representation:

```
<action>
    <host>
        <name>host.name</name>
    </host>
    <resolution_type>COPY</resolution_type>
    <job href="/api/jobs/75486b97-ee5b-4a61-af6e-960b874985fc"
```

```
id="75486b97-ee5b-4a61-af6e-960b874985fc"/>
    <status>
        <state>complete</state>
    </status>
</action>
```

- **Status Conflict**

  This conflict is resolved by either enabling or disabling a hook in a cluster.

Report a bug

Chapter 9. Hosts

The **hosts** collection provides information about the hosts in a Red Hat Storage environment. An API user accesses this information through the **rel="hosts"** link obtained from the entry point URI (see Chapter 5, *Entry Point*).

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

Table 9.1. Host elements

Element	Type	Description	Properties
link rel="nics"	relationship	A link to the **nics** sub-collection for host network interfaces.	
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. See Chapter 7, *Common Features*.	
link rel="statistics"	relationship	A link to the **statistics** sub-collection for host statistics.	
address	string	The IP address or hostname of the host.	
status	See below	The host status.	
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.	
iscsi	complex	The SCSI **initiator** for the host.	
cpu	complex	Statistics for the host CPU. Includes sub-elements for the CPU's **name**, **topology cores=**, **topology sockets=** and **speed**. The **topology cores=** aggregates the total cores while the **topology sockets=** aggregates the total physical CPUs.	
version major= minor=	complex	The compatibility level of the host.	
root_password	string	The root password of this host, by convention only included in the client-provided host representation on creation.	

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**.

Example 9.1. An XML representation of a host

```
<hosts>
        <host href="/api/hosts/eaf4af64-e51c-4e72-a2cf-bdb6076a9e56"
id="eaf4af64-e51c-4e72-a2cf-bdb6076a9e56">
            <actions>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/forceselectspm" rel="forceselectspm"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/iscsilogin" rel="iscsilogin"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/iscsidiscover" rel="iscsidiscover"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/commitnetconfig" rel="commitnetconfig"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/approve" rel="approve"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/fence" rel="fence"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/install" rel="install"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/activate" rel="activate"/>
                <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/deactivate" rel="deactivate"/>
            </actions>
            <name>host1</name>
            <comment></comment>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/storage" rel="storage"/>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics" rel="nics"/>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/tags" rel="tags"/>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/permissions" rel="permissions"/>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/statistics" rel="statistics"/>
            <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/hooks" rel="hooks"/>
            <address>10.70.43.145</address>
            <certificate>
                <organization>Test</organization>
                <subject>O=Test,CN=10.70.43.145</subject>
            </certificate>
            <status>
                <state>up</state>
            </status>
            <cluster href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59" id="419590b8-5aa0-473b-9651-aa41f1372c59"/>
            <port>54321</port>
            <type>rhel</type>
             <storage_manager priority="5">false</storage_manager>
            <version major="4" minor="14" build="7" revision="2"
full_version="vdsm-4.14.7.2-1.el6rhs"/>
            <hardware_information>
```

```
                <manufacturer>Red Hat</manufacturer>
                <version>6Server-6.4.0.4.el6</version>
                <serial_number>4C4C4544-004E-5A10-8031-
B1C04F4E5631_bc:30:5b:f2:bc:60</serial_number>
                <product_name>RHEV Hypervisor</product_name>
                <uuid>f70d244f-5fd9-4106-965d-a3d1bb1f0b3b</uuid>
                <family>Red Hat Enterprise Linux</family>
            </hardware_information>
            <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.redhat:4c5810adce86</initiator>
             </iscsi>
            <ssh>
                <port>22</port>

<fingerprint>7b:e5:04:1f:df:b4:73:ed:4f:c2:7a:97:a7:5a:93:98</fingerprin
t>
            </ssh>
            <cpu>
                <topology sockets="1" cores="1" threads="1"/>
                <name>Intel Xeon E312xx (Sandy Bridge)</name>
                <speed>2200</speed>
            </cpu>
            <memory>2103443456</memory>
            <max_scheduling_memory>1698693120</max_scheduling_memory>
            <summary>
                <active>0</active>
                <migrating>0</migrating>
                <total>0</total>
            </summary>
            <os type="RHEL">
                <version full_version="6Server - 6.5.0.1.el6"/>
            </os>
            <libvirt_version major="0" minor="10" build="2"
revision="0" full_version="libvirt-0.10.2-18.el6_4.15"/>
        </host>
</hosts>
```

Creation of a new host requires the **name**, **address** and **root_password** elements. See Section 7.2.4, “Creating a Resource in a Collection” for more information.

Example 9.2. Creating a host

```
POST /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>
  <cluster>cluster_name</cluster>
</host>
```

New host creation applies only to the addition of Red Hat Enterprise Linux hosts.

The **root_password** element is only included in the client-provided initial representation and is not exposed in the representations returned from subsequent requests.

The **name** and **cluster** elements are updatable post-creation. See Section 7.3.2, “Updating a Resource” for more information.

Example 9.3. Updating a host

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<host>
  <name>host3</name>
</host>
```

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

Example 9.4. Removing a host

```
DELETE /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3

HTTP/1.1 204 No Content
```

Report a bug

9.1. Network Interface Sub-Collection

The **nics** sub-collection represents a host's physical network interfaces. Each **host_nic** element in the representation acts as a network interface and contains the following elements:

Note

The icons used in the properties column of this table are described in Table 7.1, "Element property icons"

Table 9.2. 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="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=	complex	The IP level configuration of 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**.	
speed	integer	The network interface speed in bits per second.	
status	enumerated	The link status for the network interface. These states are listed in **host_nic_states** under **capabilities**.	
vlan id	integer	The VLAN which this interface represents.	
bonding	complex	A list of **options** and **slave** NICs for bonded interfaces.	[c]

[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 9.5. An XML representation of a network interface on a host

```
<host_nic href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-ea6045702b2a" id="4800fa31-
08f0-40bb-9964-ea6045702b2a">
    <actions>
        <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-ea6045702b2a/attach"
rel="attach"/>
        <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-ea6045702b2a/detach"
rel="detach"/>
    </actions>
    <name>eth0</name>
    <link href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-ea6045702b2a/statistics"
rel="statistics"/>
    <host href="/api/hosts/eaf4af64-e51c-4e72-a2cf-bdb6076a9e56"
id="eaf4af64-e51c-4e72-a2cf-bdb6076a9e56"/>
    <network href="/api/networks/00000000-0000-0000-0000-000000000009"
id="00000000-0000-0000-0000-000000000009"/>
    <mac address="00:15:1e:00:9d:4c"/>
    <ip address="10.70.43.145" netmask="255.255.252.0"
gateway="10.70.43.254"/>
    <boot_protocol>dhcp</boot_protocol>
    <status>
        <state>up</state>
    </status>
    <mtu>1500</mtu>
    <bridged>true</bridged>
    <custom_configuration>false</custom_configuration>
</host_nic>
```

An API user creates only bonded interfaces (see Section 9.1.1, “Bonded Interfaces”). All other network interfaces contain updatable **network**, **ip** and **boot_protocol** elements using a **PUT** request.

When adding a new network interface, the **name** and **network** elements are required. Identify the **network** element with the **id** attribute or **name** element.

An API user modifies a network interface with a **PUT** request.

```
PUT /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/nics/
e8f02fdf-3d7b-4135-86e1-1bf185570cd8 HTTP/1.1
Accept: application/xml
Content-type: application/xml

<nic>
    <ip address="192.168.0.129" netmask="255.255.255.0"
gateway="192.168.0.1"/>
</nic>
```

An API user removes a network interface with a **DELETE** request.

```
DELETE /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/nics/
```

```
e8f02fdf-3d7b-4135-86e1-1bf185570cd8 HTTP/1.1

HTTP/1.1 204 No Content
```

Report a bug

9.1.1. Bonded Interfaces

A bonded interface is represented as a **host_nic** resource containing a **bonding** element.

Table 9.3. 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 **POST**ing to a **host_nic** with bonding options and slave interfaces. The **name**, **network** and **bonded** elements are required when creating a new bonded interface. Either the **id** or **name** elements identify the **network** and slave **host_nic**s.

Example 9.6. Creating a bonded interface

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/nics HTTP/1.1
Accept: application/xml
Content-Type: application/xml

<host_nic>
    <name>bond4</name>
    <network id="e657d631-657d-42bb-a536-73501a085d85"/>
    <bonding>
        <options>
            ...
        </options>
        <slaves>
            <host_nic id="eb14e154-5e73-4f7f-bf6b-7f52609d94ec"/>
            <host_nic id="6aede5ca-4c54-4b37-a81b-c0d6b53558ea"/>
        </slaves>
    </bonding>
</host_nic>
```

Important

bond0, **bond1**, **bond2**, **bond3** and **bond4** are the only valid names for a bonded interface.

Like other resources, a **DELETE** request to a bonded interface URI deletes it.

Important

Changes to bonded interface configuration must be explicitly committed. See Section 9.3.5, "Commit Network Configuration Action".

Report a bug

9.1.2. 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 9.4. 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 9.5. 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.
errors.total.rx	Total errors from receiving data.
errors.total.tx	Total errors from transmitting data.

Example 9.7. An XML representation of a host's network interface statistics sub-collection

```
<statistics>
    <statistic href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-
ea6045702b2a/statistics/ecd0559f-e88f-3330-94b4-1f091b0ffdf7"
```

```
id="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>
        <host_nic href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/nics/4800fa31-08f0-40bb-9964-ea6045702b2a" id="4800fa31-
08f0-40bb-9964-ea6045702b2a"/>
    </statistic>
    ...
</statistics>
```

Note

This **statistics** sub-collection is read-only.

Report a bug

9.1.3. Attach Action

A host network interface is attached to a network, indicating the given network is accessible over the interface. Either the **id** or **name** elements identify the **network**.

Example 9.8. Action to attach a host network interface to a network

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/nics/e8f02fdf-3d7b-
4135-86e1-1bf185570cd8/attach HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <network id="e657d631-657d-42bb-a536-73501a085d85"/>
</action>
```

Important

This networking configuration change must be explicitly committed. See Section 9.3.5, “Commit Network Configuration Action”.

Report a bug

9.1.4. Detach Action

Detach an interface from a network. Either the **id** or **name** elements identify the **network**.

Example 9.9. Action to detach a host network interface to a network

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/nics/e8f02fdf-3d7b-
4135-86e1-1bf185570cd8/detach HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <network id="e657d631-657d-42bb-a536-73501a085d85"/>
</action>
```

Important

This networking configuration change must be explicitly committed. See Section 9.3.5, "Commit Network Configuration Action".

Report a bug

9.2. Statistics Sub-Collection

Each host resource exposes a **statistics** sub-collection for host-specific statistics. Each **statistic** contains the following elements:

Table 9.6. 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 9.7. 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.buffers	I/O buffers in bytes.

Name	Description
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 9.10. An XML representation of the host's statistics sub-collection

```
<statistics>
    <statistic href="/api/hosts/eaf4af64-e51c-4e72-a2cf-
bdb6076a9e56/statistics/7816602b-c05c-3db7-a4da-3769f7ad8896"
id="7816602b-c05c-3db7-a4da-3769f7ad8896">
        <name>memory.total</name>
        <description>Total memory</description>
        <values type="INTEGER">
            <value>
                <datum>2103443456</datum>
            </value>
        </values>
        <type>GAUGE</type>
        <unit>BYTES</unit>
        <host href="/api/hosts/eaf4af64-e51c-4e72-a2cf-bdb6076a9e56"
id="eaf4af64-e51c-4e72-a2cf-bdb6076a9e56"/>
    </statistic>
</statistics>
```

Note

A host's **statistics** sub-collection is read-only.

Report a bug

9.3. Actions

The following sections describe the actions associated with **host** resources.

The API contains a number of possible actions for hosts: **install**, **activate**, **fence**, **deactivate**, **approve**, **iscsilogin**, **iscsidiscover** and **commitnetconfig**.

Report a bug

9.3.1. Install Action

Install VDSM and related software on the host. The host type defines additional parameters for the action.

- **Red Hat Enterprise Linux host** - This host type requires a **root_password** element that refers to the password for the host's **root** user.

Example 9.11. Action to install VDSM to a Red Hat Enterprise Linux host

```
POST /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>
```

Report a bug

9.3.2. Activate Action

Activate the host for use, such as creating bricks.

Example 9.12. Action to activate a host

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/activate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

Report a bug

9.3.3. Fence Action

An API user controls a host's power management device with the **fence** action. The **capabilities** lists available **fence_type** options.

Example 9.13. Action to fence a host

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/fence
Accept: application/xml
Content-Type: application/xml

<action>
    <fence_type>start</fence_type>
</action>
```

Report a bug

9.3.4. Deactivate Action

Deactivate the host to perform maintenance tasks.

Example 9.14. Action to deactivate a host

```
POST /api/hosts/2ab5e1da-b726-4274-bbf7-0a42b16a0fc3/deactivate HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action/>
```

Report a bug

9.3.5. Commit 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 9.15. Action to commit network configuration

```
POST /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 Red Hat Storage Console 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.

Report a bug

Chapter 10. Volumes

The **volumes** collection provides information about volumes in a Red Hat Storage environment.

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

Table 10.1. Volume elements

Element	Type	Description	Properties
volumeName	relationship	Name of the volume to be created.	
volumeType	relationship	DISTRIBUTE,REPLICATE, DISTRIBUTED_REPLICATE, STRIPE, DISTRIBUTED_STRIPE.	
replicaCount	complex	replicaCount is mandatory if volumeType is REPLICATE or DISTRIBUTED_REPLICATE	
stripeCount	complex	stripeCount is mandatory if volumeType is STRIPE or DISTRIBUTED_STRIPE	
bricks	complex	list of bricks of a volume. You can add/remove bricks to/from a volume.	
options	complex	list of options of the volume	

Example 10.1. An XML representation of a volume

```
  <gluster_volume href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50"
id="6d4bc7ed-4278-45e1-973e-e9a5e061de50">
    <actions>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/rebalance" rel="rebalance"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/stoprebalance" rel="stoprebalance"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/resetoption" rel="resetoption"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/resetalloptions" rel="resetalloptions"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/setoption" rel="setoption"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/start"
rel="start"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/stop"
rel="stop"/>
```

```
    </actions>
    <name>newVol</name>
    <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/bricks"
rel="bricks"/>
    <cluster href="/api/clusters/419590b8-5aa0-473b-9651-aa41f1372c59"
id="419590b8-5aa0-473b-9651-aa41f1372c59"/>
    <volume_type>distribute</volume_type>
    <transport_types>
        <transport_type>tcp</transport_type>
    </transport_types>
    <replica_count>0</replica_count>
    <stripe_count>0</stripe_count>
    <options/>
    <status>
        <state>down</state>
    </status>
</gluster_volume>
```

Report a bug

10.1. Creating a Volume

Creation of a new volume requires the **volumeName**, **volumeType,transportType** and **brick** elements. The API creates a new volume with a **POST** request to the URI containing a representation of the new volume

.

Example 10.2. Creating a volume

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

<gluster_volume>
  <name>data</name>
  <volume_type>distribute</volume_type>
  <bricks>
    <brick>
      <server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
      <brick_dir>/export/data/brick1</brick_dir>
    </brick>
    <brick>
      <server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
      <brick_dir>/export/data/brick2</brick_dir>
    </brick>
  </bricks>
</gluster_volume>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] --cacert [CERT] https://[RHSC
HOST]/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes -d "
<gluster_volume><name>data</name><volume_type>DISTRIBUTE</volume_type>
<bricks><brick><server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
<brick_dir>/export/data/brick1</brick_dir></brick><brick>
<server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
<brick_dir>/export/data/brick2</brick_dir></brick></bricks>
</gluster_volume>"
```

The API returns the following representation of the newly created volume resource:

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

<gluster_volume href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554"
id="83101900-2f12-4855-838e-36b8a9e04554">
    <name>data</name>
    <actions>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/start"
rel="start"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/stop"
rel="stop"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/setOption" rel="setOption"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/rebalance" rel="rebalance"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/resetOption" rel="resetOption"/>
        <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/resetAllOptions" rel="resetAllOptions"/>
    </actions>
    <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks"
rel="bricks"/>
    <cluster href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95"
id="99408929-82cf-4dc7-a532-9d998063fa95"/>
    <volume_type>distribute</volume_type>
    <transport_types>
        <transport_type>TCP</transport_type>
    </transport_types>
    <replica_count>0</replica_count>
    <stripe_count>0</stripe_count>
    <options/>
    <state>DOWN</state>
</gluster_volume>
```

Report a bug

10.2. Listing Volumes

A listing of volumes in a cluster is obtained by issuing a **GET** request on the cluster **URI**.

Example 10.3. Listing Volumes

```
GET /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes
HTTP/1.1
Accept: application/xml
```

cURL command:

curl -X GET -H "Accept: application/xml" -u *[USER:PASS]* https://*[RHSC Host]*/api/clusters/*99408929-82cf-4dc7-a532-9d998063fa95*/glustervolumes

The API returns the following representation:

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

<gluster_volume href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50"
id="6d4bc7ed-4278-45e1-973e-e9a5e061de50">
    <actions>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/rebalance" rel="rebalance"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/stoprebalance" rel="stoprebalance"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/resetoption" rel="resetoption"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/resetalloptions" rel="resetalloptions"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-
e9a5e061de50/setoption" rel="setoption"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/start"
rel="start"/>
        <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/stop"
rel="stop"/>
    </actions>
    <name>newVol</name>
    <link href="/api/clusters/419590b8-5aa0-473b-9651-
aa41f1372c59/glustervolumes/6d4bc7ed-4278-45e1-973e-e9a5e061de50/bricks"
rel="bricks"/>
    <cluster href="/api/clusters/419590b8-5aa0-473b-9651-aa41f1372c59"
id="419590b8-5aa0-473b-9651-aa41f1372c59"/>
    <volume_type>distribute</volume_type>
```

```
    <transport_types>
        <transport_type>tcp</transport_type>
    </transport_types>
    <replica_count>0</replica_count>
    <stripe_count>0</stripe_count>
    <options/>
    <status>
        <state>down</state>
    </status>
</gluster_volume>
```

Report a bug

10.3. Managing Volumes

The API starts, stops, re-balances, and set volume option of your volume in your cluster by issuing a **POST** request to the **URI**.

Report a bug

10.3.1. Starting a Volume

The API starts a volume with a **POST** request to the **volumes** URI.

Example 10.4. Starting a Volume

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/start
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
   [ <force>true</force> ]
</action>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC-
Host]/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/start -
d "<action/>"
```

The API returns the following representation:

```
<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

10.3.2. Stopping a Volume

Stopping of a volume requires a **POST** request.

Example 10.5. Stopping a Volume

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/stop
HTTP/1.1
Accept: application/xml
Content-type: application/xml
<action>
   [ <force>true</force> ]
</action>

<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC-
Host]/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/stop -
d "<action/>"
```

The API returns the following representation:

```
<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

10.3.3. Removing a Volume

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

Example 10.6. Removing a Volume

```
DELETE /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/ceb59bbb-173d-4a5a-9b92-0189a17eae27
HTTP/1.1

HTTP/1.1 204 No Content
```

cURL command:

```
curl -X DELETE -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC-Host]/api/clusters/99408929-
82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554"
```

Report a bug

10.3.4. Setting a Volume Option

The API sets a volume option with a **POST** request to the volume **URI**.

Example 10.7. Setting a Volume Option

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/setOption HTTP/1.1
Accept: application/xml
Content-type: application/xml

<action>
    <option name="key" value="value" />
</action>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS]https://[RHSC-Host]/api/clusters/99408929-
82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/setOption -d "<action><option name="key" value="value" />
</action>"
```

Report a bug

10.3.5. Resetting a Volume Option

Resetting volume options requires a **POST** request.

Example 10.8. Resetting a Volume Option

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/resetOption
HTTP/1.1
Accept: application/xml
Content-type: application/xml
<action>
    <option name="key" value="value"/>
    <force>true</force>
</action>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC-Host]/api/clusters/99408929-
82cf-4dc7-a532- 9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/resetOption -d "<action><option name="key" value="value"/>
<force>true</force></action>"
```

The API returns the following representation:

```
<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

10.3.6. Resetting all Volume Options

You can reset all volume options with a single **POST** request.

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/ceb59bbb-173d-4a5a-9b92-
0189a17eae27/resetAllOptions HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC-Host]/api/clusters/99408929-
82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/resetAllOptions -d "<action/>"
```

The API returns the following representation:

```
<action>
    <status>
        <state>complete</state>
    </status>
</action>
```

Report a bug

10.3.7. Rebalancing Volume

You can rebalance the volume by issuing a **POST** request. The status of rebalance operation can be monitored by job id returned by this API.

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/ceb59bbb-173d-4a5a-9b92-
0189a17eae27/rebalance -d "<action/>"
```

```
HTTP/1.1
Accept: application/xml
Content-type: application/xml
```

cURL command:

curl -X POST -H "Accept: application/xml" -H "Content-Type: application/xml" -u *[USER:PASS]* https://*[RHSC-Host]*/api/clusters/44571c63-a110-4fba-9a8c-7b30446ba8bf/glustervolumes/380474af-360b-48fa-a210-9c08b913ffa3/rebalance -d "<action/>"

The API returns the following representation:

```
<action>
    <job href="/api/jobs/0937afb0-2c21-4834-8fee-70e8982b8a8e"
id="0937afb0-2c21-4834-8fee-70e8982b8a8e"/>
    <status>
        <state>complete</state>
    </status>
</action>
```

The status of rebalance is monitored using the job ID and its corresponding step ID by looking at the re-balancing step.

```
/api/jobs/<job_id>/steps
/api/jobs/<job_id>/steps/<step_id> --for step type rebalance
/api/jobs/<job_id>/steps/<step_id>/statistics -- for detail information
of rebalance status per node
```

Example 10.9. Monitoring Rebalance

```
<statistics>
    <statistic href="/api/hosts/93fb8339-3888-44de-8050-
e1166714e186/statistics/efca817c-2c70-3c82-b5b4-cff3dc24acbd"
id="efca817c-2c70
-3c82-b5b4-cff3dc24acbd">
        <name>files.moved</name>
        <description>Number of files moved</description>
        <values type="INTEGER">
            <value>
                <datum>0</datum>
            </value>
        </values>
        <type>COUNTER</type>
        <unit>NONE</unit>
        <host href="/api/hosts/93fb8339-3888-44de-8050-e1166714e186"
id="93fb8339-3888-44de-8050-e1166714e186"/>
        <step href="/api/jobs/0937afb0-2c21-4834-8fee-
70e8982b8a8e/steps/1560867e-7460-49d4-8bdb-6444a6ffef3a" id="1560867e-
7460-49d4-8
bdb-6444a6ffef3a"/>
    </statistic>
...
</statistics>
```

You can stop a rebalance operation by issuing a POST request.

curl -X POST -H "Accept: application/xml" -H "Content-Type: application/xml" -u *[USER:PASS]* https://*[RHSC-Host]*/api/clusters/44571c63-a110-4fba-9a8c-7b30446ba8bf/glustervolumes/380474af-360b-48fa-a210-9c08b913ffa3/stoprebalance -d "<action/>"

Report a bug

10.4. Managing Bricks

This section describes the brick operations of the Red Hat Storage Console API such as adding and removing them from a volume.

Report a bug

10.4.1. Listing of Bricks

A listing of bricks in a volume is obtained by issuing a **GET** request on the volume **URI**.

```
GET /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks
HTTP/1.1
Accept: application/xml
```

The API returns the following representation:

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

<bricks>
    <brick href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks/29311ecf-d8db-4912-902b-b31dba8803d2" id="29311ecf-d8db-4912-902b-b31dba8803d2">
        <actions>
            <link href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks/29311ecf-d8db-4912-902b-b31dba8803d2/replace" rel="replace"/>
        </actions>
        <gluster_volume href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554" id="83101900-2f12-4855-838e-36b8a9e04554"/>
        <server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
        <brick_dir>/tmp/data-brick1</brick_dir>
        <state>UP</state>
    </brick>
    <brick href="/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks/33f04f44-c78c-4c76-a23c-fe9d750c2fd3" id="33f04f44-c78c-4c76-a23c-fe9d750c2fd3">
        <actions>
```

```
            <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/bricks/33f04f44-c78c-4c76-a23c-fe9d750c2fd3/replace"
rel="replace"/>
        </actions>
        <gluster_volume href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554"
id="83101900-2f12-4855-838e-36b8a9e04554"/>
        <server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
        <brick_dir>/tmp/data-brick2</brick_dir>
        <state>UP</state>
    </brick>
</bricks>
```

Report a bug

10.4.2. Adding a Brick

The API adds a new brick to a volume in your cluster with a **POST** request to its **URI**.

```
POST /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks
HTTP/1.1
Accept: application/xml
Content-type: application/xml

<bricks>
  <brick>
<server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
    <brick_dir>/export/data/brick3</brick_dir>
  </brick>
</bricks>
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" -u [USER:PASS] https://[RHSC HOST]/api/clusters/99408929-
82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/bricks -d "<bricks><brick><server_id>fcb46b88-f32e-11e1-918a-
0050568c4349</server_id><brick_dir>/export/data/brick3</brick_dir></brick>
</bricks>"
```

The API returns the following representation:

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

<bricks>
    <brick href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/bricks/a2a496fc-9df0-446f-b632-da808d65d501" id="a2a496fc-
9df0-446f-b632-da808d65d501">
        <actions>
            <link href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
```

```
36b8a9e04554/bricks/a2a496fc-9df0-446f-b632-da808d65d501/replace"
rel="replace"/>
        </actions>
        <gluster_volume href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554"
id="83101900-2f12-4855-838e-36b8a9e04554"/>
        <server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
        <brick_dir>/export/data/brick3</brick_dir>
        <state>UP</state>
    </brick>
</bricks>
```

Report a bug

10.4.3. Removing a Brick

The API removes a brick(s) from a volume with a **DELETE** request sent to its **URI**. Before removing a brick, data must be migrated to ensure there is no data loss with a **POST** request sent to its **URI**.

```
POST /api/clusters/44571c63-a110-4fba-9a8c-
7b30446ba8bf/glustervolumes/380474af-360b-48fa-a210-
9c08b913ffa3/bricks/migrate -d "<action><bricks><brick>
<name>host:brick_dir</name></brick></bricks></action>"
```

cURL command:

```
curl -X POST -H "Accept: application/xml" -H "Content-Type:
application/xml" [USER:PASS] https://[RHSC HOST]/api/clusters/44571c63-
a110-4fba-9a8c-7b30446ba8bf/glustervolumes/380474af-360b-48fa-a210-
9c08b913ffa3/bricks/migrate -d "<action><bricks><brick>
<name>host:brick_dir</name></brick></bricks></action>"
```

The API returns the following representation:

```
<action>
    <bricks>
        <brick>
            <name>host:brick_dir</name>
        </brick>
    </bricks>
    <job href="/api/jobs/5d070f64-2f9b-44d7-8c44-64829355422d"
id="5d070f64-2f9b-44d7-8c44-64829355422d"/>
    <status>
        <state>complete</state>
    </status>
</action>
```

The status of remove brick (with data migration) is monitored using the job ID and its corresponding step ID by looking at the remove brick step.

```
/api/jobs/<job_id>/steps
/api/jobs/<job_id>/steps/<step_id> --for step type remove brick
/api/jobs/<job_id>/steps/<step_id>/statistics -- for detail information
of remove brick
```

Now, you can delete the migrated bricks by issuing a **DELETE** request for a brick(s).

```
DELETE /api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95/glustervolumes/83101900-2f12-4855-838e-36b8a9e04554/bricks
```

cURL command:

```
curl -X DELETE -u [USER:PASS] https://[RHSC HOST]/api/clusters/99408929-
82cf-4dc7-a532-9d998063fa95/glustervolumes/83101900-2f12-4855-838e-
36b8a9e04554/bricks -d "<bricks><brick><name>host:brick_dir</brick>
</bricks>"
```

Report a bug

Chapter 11. Groups

The **groups** collection contains imported groups from directory services.

A **group** resource contains a set of elements.

Table 11.1. 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.

Example 11.1. An XML representation of a group resource

```
<group id="85bf8d97-273c-4a5c-b801-b17d58330dab"
  href="/api/groups/85bf8d97-273c-4a5c-b801-b17d58330dab">
    <name>Everyone</name>
    <link rel="tags"
      href="/api/groups/85bf8d97-273c-4a5c-b801-b17d58330dab/tags"/>
    <link rel="permissions"
      href="/api/groups/85bf8d97-273c-4a5c-b801-
b17d58330dab/permissions"/>
    <link rel="roles"
      href="/api/groups/85bf8d97-273c-4a5c-b801-b17d58330dab/roles"/>
</group>
```

Report a bug

Chapter 12. Roles

The **rel="roles"** link obtained from the entry point URI (see Chapter 5, *Entry Point*) provides access to a static set of system roles.

Each individual **role** element contains the following:

Table 12.1. 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 Storage Console environment.	
administrative	Boolean: true or false	Defines the role as administrative-only.	

Example 12.1. An XML representation of the roles collection

```
<roles>
    <role id="00000000-0000-0000-0000-000000000001"
      href="/api/roles/00000000-0000-0000-0000-000000000001">
        <name>SuperUser</name>
        <description>Roles management administrator</description>
        <link rel="permits"
          href="/api/roles/00000000-0000-0000-0000-
000000000001/permits"/>
        <mutable>false</mutable>
        <administrative>true</administrative>
    </role>
    <role id="00000000-0000-0000-0001-000000000001"
      href="/api/roles/00000000-0000-0000-0001-000000000001">
        <name>RHSCUser</name>
        <description>RHSC user</description>
        <link rel="permits"
          href="/api/roles/00000000-0000-0000-0001-
000000000001/permits"/>
        <mutable>false</mutable>
        <administrative>false</administrative>
    </role>
  </roles>
```

Creation of a role requires values for **name**, **administrative** and a list of initial **permits**. See Section 7.2.4, “Creating a Resource in a Collection” for more information.

Example 12.2. Creating a role

```
POST /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>
```

The **name**, **description** and **administrative** elements are updatable post-creation. See Section 7.3.2, “Updating a Resource” for more information.

Example 12.3. Updating a role

```
PUT /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>
```

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

Example 12.4. Removing a role

```
DELETE /api/roles/8de42ad7-f307-408b-80e8-9d28b85adfd7

HTTP/1.1 204 No Content
```

Report a bug

12.1. 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 12.5. Listing a role's permits

```
GET /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="/api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9/permits/1">
        <name>create_vm</name>
        <administrative>false</administrative>
        <role id="b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"
          href="/api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"/>
    </permit>
    ...
</permits>
```

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 12.6. Assign a permit to a role

```
POST /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="/api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9/permits/1">
        <name>create_vm</name>
        <administrative>false</administrative>
        <role id="b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"
          href="/api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9"/>
    </permit>
</permits>
```

Remove a **permit** from a role with a **DELETE** request to the **permit** resource.

Example 12.7. Remove a permit from a role

```
DELETE /api/roles/b67dfbe2-0dbc-41e4-86d3-a2fbef02cfa9/permits/

HTTP/1.1 204 No Content
```

Report a bug

Chapter 13. Users

Users are exposed in a top-level collection and are referenced with the **rel="users"** link. Individual **user** elements contain the following:

Table 13.1. 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.	

Example 13.1. An XML representation of a user resource

```
GET /api/users HTTP/1.1
Accept: application/xml

<user id="225f15cd-e891-434d-8262-a66808fcb9b1"
  href="/api/users/225f15cd-e891-434d-8262-a66808fcb9b1">
    <name>Admin</name>
    <actions/>
    <link rel="roles"
      href="/api/users/225f15cd-e891-434d-8262-a66808fcb9b1/roles"/>
    <link rel="tags"
      href="/api/users/225f15cd-e891-434d-8262-a66808fcb9b1/tags"/>
    <domain>domain.example.com</domain>
    <logged_in>false</logged_in>
    <user_name>admin@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>
```

The API adds an existing directory service user to the Red Hat Storage Console 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 13.2. Adding a user from directory service and assigning two roles

```
POST /api/users HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<user>
    <user_name>joe@domain.example.com</user_name>
    <roles>
        <role>
            <name>RHSC User</name>
        </role>
        <role id="00000000-0000-0000-0001-000000000003"/>
    </roles>
</user>
```

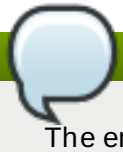
The new user is identified either by Red Hat Storage Console user ID or via the directory service user principal name (UPN). The user ID format reported from the directory service domain might be different to the expected Red Hat Storage Console format, such as in LDIF, [3]the ID has the opposite byte 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 Storage Console 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.

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 **RHSCUser** role to the role assignments for a particular user.



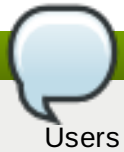
Note

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 13.3. Adding roles to a user

```
POST /api/users/225f15cd-e891-434d-8262-a66808fcb9b1/roles HTTP/1.1
Content-Type: application/xml
Accept: application/xml

<role>
    <name>RHSCUser</name>
</role>
```



Note

Users are not updated with the **PUT** verb. The only changes allowed post-creation are in the user's role assignments.

The API removes users from the Red Hat Storage Console database with a **DELETE** request on the **users** collection. The directory service domain remains unchanged after such a deletion.

Report a bug

[3] The LDAP Data Interchange Format is described in RFC 2849.

Chapter 14. Events

The **rel="events"** link obtained from the entry point URI accesses the **events** collection and lists system events from Red Hat Storage Console.

Table 14.1. 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.
user id	GUID	The identification code for the user who triggered the event.

Example 14.1. An XML representation of the events collection

```
<events>
    <event href="/api/events/767" id="767">
        <description>User rhcadmin logged in.</description>
        <code>30</code>
        <severity>normal</severity>
        <time>2014-01-29T14:42:55.041+05:30</time>
        <user href="/api/users/fdfc627c-d875-11e0-90f0-83df133b58cc"
id="fdfc627c-d875-11e0-90f0-83df133b58cc"/>
        <origin>oVirt</origin>
        <custom_id>-1</custom_id>
        <flood_rate>30</flood_rate>
 </event>
    ...
</events>
```

In addition to **user**, an **event** representation also contains a set of XML element relationships to resources relevant to the event.

Example 14.2. An XML representation of a volume start event

```
<event href="/api/events/192" id="192">
        <description>Gluster Volume data started.</description>
        <code>4004</code>
        <severity>normal</severity>
        <time>2012-09-13T20:59:22.137-04:00</time>
        <user href="/api/users/fdfc627c-d875-11e0-90f0-83df133b58cc"
id="fdfc627c-d875-11e0-90f0-83df133b58cc"/>
        <cluster href="/api/clusters/99408929-82cf-4dc7-a532-
9d998063fa95" id="99408929-82cf-4dc7-a532-9d998063fa95"/>
    </event>
```

This example provides an XML element representation of starting a volume on the **Default** cluster .

Note

The **events** collection is read-only.

Report a bug

14.1. Searching Events

The **events** collection provides search queries similar to other resource collections (see Section 7.2.3, “Searching Collections with Queries”). 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** argument added to the URI after the query. This **from** argument references an event **id** code.

Example 14.3. Searching from an event

```
GET /api/events?search=type%3D30&from=1012 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="/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="/api/users/80b71bae-98a1-11e0-8f20-525400866c73"/>
    </event>
    <event id="1016" href="/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="/api/users/80b71bae-98a1-11e0-8f20-525400866c73"/>
    </event>
    <event id="1014" href="/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="/api/users/80b71bae-98a1-11e0-8f20-525400866c73"/>
    </event>
</events>
```

Report a bug

14.2. Paginating Events

A storage 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 storage 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 14.4. Paginating events

This example paginates **event** resources. The URL-encoded request is:

```
GET /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 /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 /api/events?search=sortby%20time%20asc%20page%202&from=30 HTTP/1.1
Accept: application/xml
```

Report a bug

Part III. Python Software Development Kit

Chapter 15. Software Development Kit Overview

15.1. Introduction to the Red Hat Storage Software Development Kit

The software development kit provides programming language specific libraries for interacting with the REST API provided by Red Hat Storage Console. These libraries include classes to represent resources made available by the API and methods for interacting with those classes. This allows you to concentrate on writing code specific to what you are trying to achieve rather than hand crafting appropriately formatted HTTP requests and manually orchestrating their delivery.

The software development kit includes collections and methods mapping directly to all aspects of the REST API. As a result, this software development kit documentation focuses on providing examples written in the supported programming language. For a complete reference of all supported collections, resources, actions, and attributes, refer to the REST API material.

Report a bug

15.2. Software Development Kit Prerequisites

To install the software development kit you must have:

- A system with Red Hat Enterprise Linux 6, or later, installed. Both the Server and Workstation variants are supported.
- A subscription to Red Hat Storage entitlements.

When you install the **rhsc-sdk** package, the Python 2.6 interpreter will be installed if it does not already exist on the system.

Important

To run scripts that use the libraries provided by the software development kit, the **rhsc-sdk** package must be installed. As a result, the prerequisites listed here must be met both on the systems being used to develop scripts using the software development kit, and on those systems on which the scripts are intended to run.

Report a bug

15.3. Installing the Software Development Kit

Summary

The software development kit is provided by the **rhsc-sdk** package. This package includes all of the Python bindings for the Red Hat Storage Console API. To begin using the software development kit, you must install the **rhsc-sdk** package on the system that you wish to use for script development. The instructions that appear here are intended for use on a system running Red Hat Enterprise Linux 6 or later.

Procedure 15.1. Installing the Python SDK

1. Ensure that your system has the required entitlements:

- When using certificate-based Red Hat Network, you must subscribe to the **Red Hat Storage** entitlement to install the **rhsc-sdk** package.
- When using Red Hat Network classic, you must subscribe to the **Red Hat Storage Console** channel to install the **rhsc-sdk** package. Refer to the Red Hat Storage Console Installation Guide for specific channel names current to your system.

2. Ensure that you are logged in as the **root** user.
3. Install the **rhsc-sdk** package using the **yum** command.

```
# yum install rhsc-sdk
```

Result

The **rhsc-sdk** package is now installed. The **ovirtsdk** Python library is now available for use on the local system.

Report a bug

Chapter 16. Using the Software Development Kit

16.1. Connecting to the API Using Python

To connect to the REST API using Python, you must create an instance of the **API** class from the *ovirtsdk.api* module. To be able to do this, it is necessary to first import the class at the start of the script:

```
from ovirtsdk.api import API
```

The constuctor of the **API** class takes a number of arguments. Supported arguments are:

url

Specifies the URL of the Manager to connect to, including the **/api** path. This parameter is *mandatory*.

username

Specifies the user name to use when connecting, in the User Principle Name (UPN) format. This parameter is *mandatory*.

password

Specifies the password for the user provided by the **username** parameter. This parameter is *mandatory*.

key_file

Specifies a PEM-formatted key file containing the private key associated with the certificate specified by **cert_file**. This parameter is optional.

cert_file

Specifies a PEM-formatted client certificate to be used for establishing the identity of the client on the server. This parameter is optional.

ca_file

Specifies the certificate file of the certificate authority for the server. This parameter is mandatory unless the **insecure** parameter is set to **True**.

port

Specifies the port to use when connecting, where it has not been provided as component of the **url** parameter. This parameter is optional.

timeout

Specifies the amount of time in seconds that is allowed to pass before a request is considered to be timed out. This parameter is optional.

persistent_auth

Specifies whether persistent authentication is enabled for this connection. Valid values are **True** and **False**. This parameter is optional and defaults to **False**.

insecure

Allows a connection via SSL without a certificate authority. Valid values are **True** and **False**, and the default is **False**. If the **insecure** parameter is set to **False** then the *ca_file* must be supplied to secure the connection.

This option should be used with caution, as it may allow man-in-the-middle (MITM) attackers to spoof the identity of the server.

filter

Specifies whether or not the user permission based filter is on or off. Valid values are **True** and **False**, and the default is **False**. If the **filter** parameter is set to **False**, the authentication credentials provided must be those of an administrative user. If the **filter** parameter is set to **True**, any user can be used and the Console will filter the actions available to the user based on their permissions.

debug

Specifies whether debug mode is enabled for this connection. Valid values are **True** and **False**. This parameter is optional.

You can communicate with multiple Red Hat Storage Console by creating and manipulating separate instances of the **ovirtsdk.API** Python class.

For a full list of methods supported by the **API** class, refer to the PyDoc output for the *ovirtsdk.api* package.

Report a bug

16.2. Listing the Public Attributes of a Resource

The **dir()** method returns a list of public variables, modules, and functions of a resource. The private attribute names start with an underscore character. To get a list of all the public attributes of a resource, use the following code snippet:

```
for name in (e for e in dir(api) if not e.startswith('_')):
  print name
```

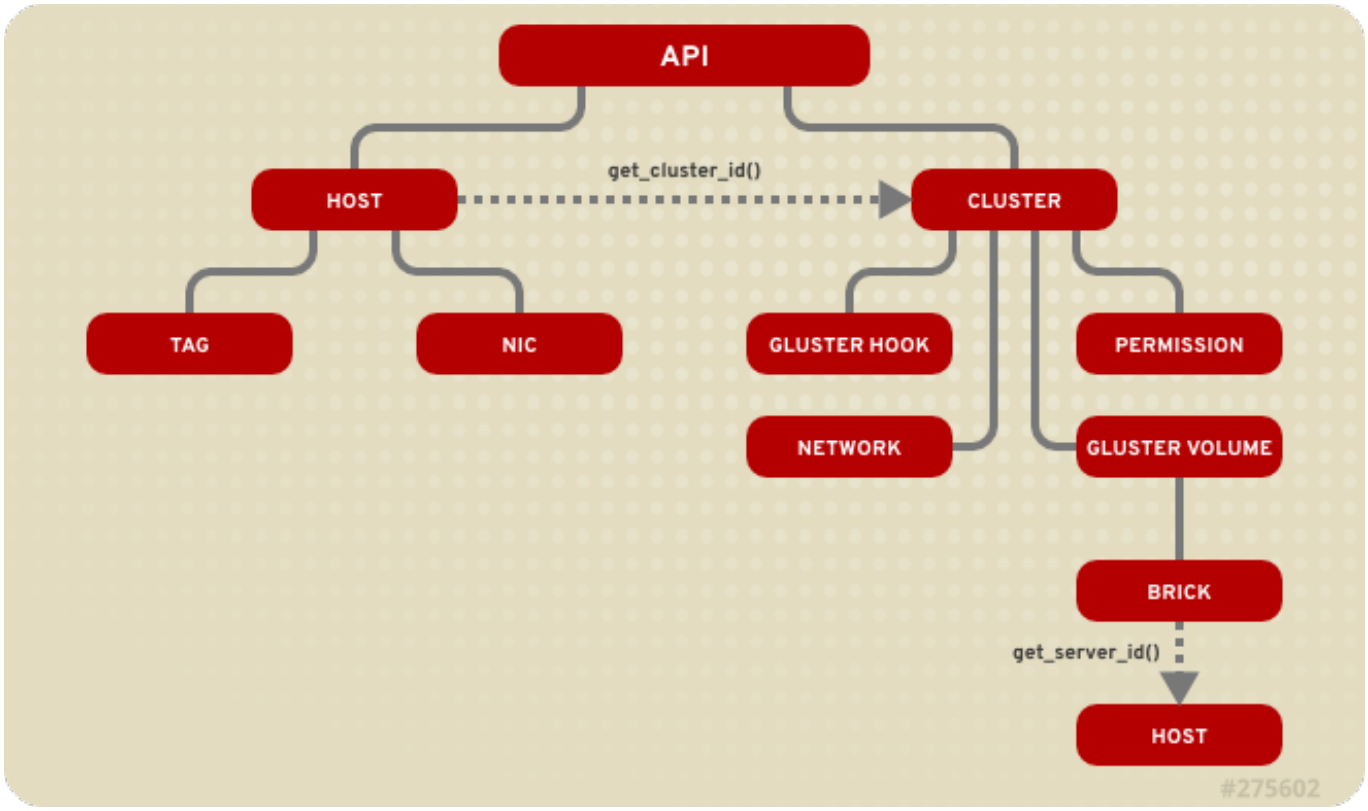


Figure 16.1. Object Hierarchy

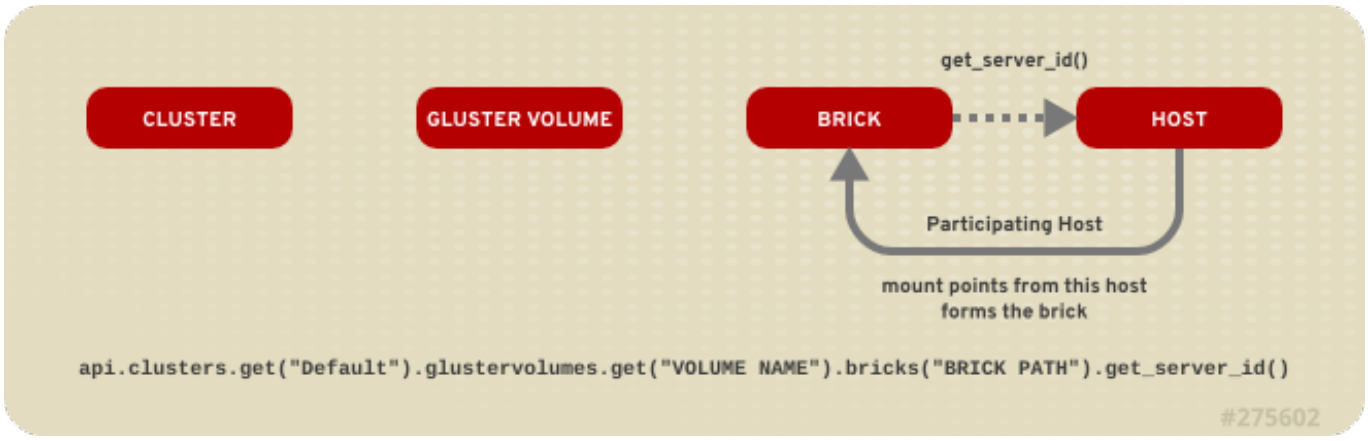


Figure 16.2. Relationship between a brick and a host

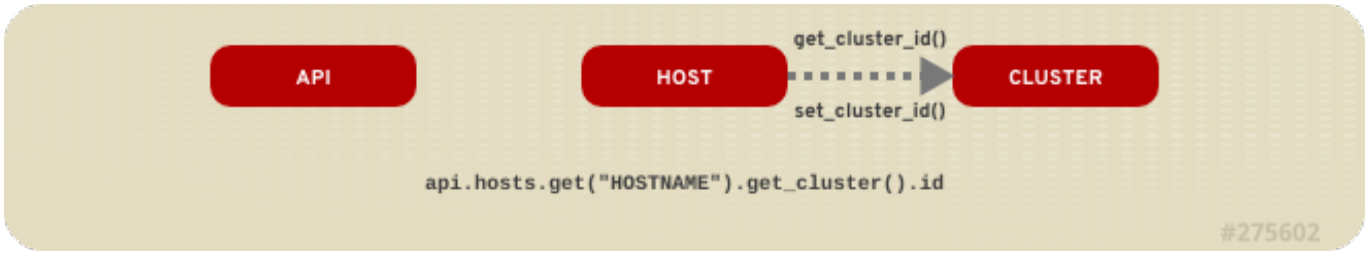


Figure 16.3. Relationship between a host and a cluster

Report a bug

16.3. Resources and Collections

The RESTful nature of the API is evident throughout the Python bindings for both philosophical and practical reasons. All RESTful APIs have two key concepts:

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 volume collection which contains all bricks in the environment. An example of a sub-collection is the **`volume.bricks`** collection which contains resources for CPUs attached to a cluster.

The interface for interacting with collections provides methods for adding resources (**`add`**), getting resources (**`get`**), and listing resources (**`list`**).

Resources

A resource in a RESTful API is an object with a fixed interface that also contains a set of attributes that are relevant to the specific type of resource being represented. The interface for interacting with resources provides methods for updating (**`update`**) and deleting (**`delete`**) resources. Additionally, some resources support actions specific to the resource type. One example of this is the **`approve`** method of **`Host`** resources.

Report a bug

16.4. Retrieving Resources from a Collection

Resources are retrieved from a collection using the **`get`** and **`list`** methods.

`get`

Retrieves a single resource from the collection. The item to retrieve is determined based on the name provided as an argument. The **`get`** method takes these arguments:

- **`name`** - The name of the resource to retrieve from the collection.
- **`id`** - The globally unique identifier (GUID) of the resource to retrieve from the collection.

`list`

Retrieves any number of resources from the collection. The items to retrieve are determined based on the criteria provided. The **`list`** method takes these arguments:

- **`**kwargs`** - A dictionary of additional arguments allowing keyword-based filtering.
- **`query`** - A query written in the same format as that used for searches executed using the Red Hat Storage Console.
- **`max`** - The maximum number of resources to retrieve.
- **`case_sensitive`** - Whether or not search terms are to be treated as case sensitive (**`True`** or **`False`**, the default is **`True`**).

> **Example 16.1. Retrieving a List of Resources in a Collection Matching a Keyword Based Filter**
>
> The example below lists all the volumes that have the volume type set to *distribute*.

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER@internal",
          password="PASS",
          ca_file="ca.crt")

    clusterName="CLUSTERNAME"
    volume_list= api.clusters.get(clusterName).glustervolumes.list(**
{"volume_type": "distribute"})

    for volume in volume_list:
        print "VolumeName:", volume.get_name()

    api.disconnect()
except Exception as ex:
    print "Unexpected error: %s" % ex
```

Report a bug

16.4.1. Retrieving a Specific Resource from a Collection

In this example, a specific resource is retrieved from a collection using the **`get`** method.

To retrieve the *Default* cluster from the *clusters* collection using the ***`name`*** parameter of the **`get`** method:

```
cl = api.clusters.get("Default")
```

This syntax is equivalent to:

```
cl = api.clusters.get(name="Default")
```

Report a bug

16.4.2. Retrieving a List of Resources from a Collection

In these examples, a list of resources is retrieved from a collection using the **`list`** method.

In the following example, a list of all resources is retrieved from the *clusters* collection. The query parameter of the list method allows the use of engine-based queries. In this way the SDK supports the use of queries in the same format as those executed in the Administration and User Portals. The query parameter is also the mechanism for providing pagination arguments while iterating through the collection.

Example 16.2. Listing All Clusters

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
```

```
        api = API (url="https://HOST,
                username="USER@DOMAIN",
                password="PASS",
                ca_file="ca.crt")

        c_list = api.clusters.list()

        for c in c_list:
            print "%s (%s)" % (c.get_name(), c.get_id())

        api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

Executing this code on a Python engine will show a list of clusters in the Red Hat Storage Console:

```
c1 (42c68146-0036-4374-a6e0-bf8d5874d890)
cl1 (d62cb3a2-1bea-4267-af3a-72aadac8ee21)
cl2 (49bc2dc6-b5fb-4da2-83ba-fad736757c14)
cl3 (0c4b3cb9-1557-4448-9fda-a9209830cec9)
Default (99408929-82cf-4dc7-a532-9d998063fa95)
tcl (cbc131f6-09f1-4f0e-8d8e-89a7c3288984)
```

In this example, the **`get_name()`** method provides the name of the cluster and the **`get_gluster_service()`** returns **`true`** or **`false`** based on glusterFS support in the cluster:

Example 16.3. Listing All Hosts

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER@DOMAIN",
          password="PASS",
          ca_file="ca.crt")

    h_list = api.hosts.list()

    for h in h_list:
        print "%s (%s)" % (h.get_name(), h.get_id())

    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

In this example, **`api.hosts.list()`** provides a dictionary list that contains the host name and the host ID.

Executing this code on a Python engine will show a list of hosts in the Red Hat Storage Console:

```
h1 (51b29199-3a7f-49e4-b0d5-92afdeea1f15)
h2 (4e8c3642-e654-456c-9dff-a240c8a43b1f)
host1 (14e3823d-d557-4635-bd9f-0c6157ab7f07)
```

Important

The **list** method of a collection is restricted to returning only as many elements as allowed by the **SearchResultsLimit** configuration key in the Red Hat Storage Console. To ensure that all records in a the **list** are returned, it is recommended to paginate through the results as illustrated in this example. Alternatively, set the **max** parameter of the **list** method to the maximum number of records that you wish to retrieve.

Example 16.4. Listing All Red Hat Storage Volumes

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER",
          password="PASS",
          ca_file="ca.crt")
    clusterName="CLUSTERNAME"

    for volume in api.clusters.get(clusterName).glustervolumes.list():
        print "VolumeName:", volume.get_name()
    api.disconnect()
except Exception as ex:
    print "Unexpected error: %s" % ex
```

Executing this code on a Python engine will show a list of Red Hat Storage Volumes in the Red Hat Storage Console:

```
VolumeName: vol1
```

Example 16.5. Listing All Users

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    api = API (url="https://HOST",
          username="USER",
          password="PASS",
          ca_file="ca.crt")
    for user in api.users.list():
        print "User name: %s (User ID: %s)" % (user.get_name(),
user.get_id())
```

```
    api.disconnect()

except Exception as ex:
    print "Unexpected error: %s" % ex
```

Including this code snippet in the program will return a list of Red Hat Storage Console users:

```
User name: admin (User ID: fdfc627c-d875-11e0-90f0-83df133b58cc)
```

Example 16.6. Listing All Roles

The **`getUsers()`** function lists all the roles in the Red Hat Storage Console.

```
def listRoles():
     """ Return list of user roles """
    roles = []
    for role in api.roles.list():
        roles.append(role)
    return roles
```

Including this code snippet in the program will return a list of roles defined in the Red Hat Storage Console:

```
for i in api.roles.list():
   print i.name
```

```
SuperUser
ClusterAdmin
HostAdmin
NetworkAdmin
GlusterAdmin
ExternalEventsCreator
ExternalTasksCreator
```

Example 16.7. Listing All Network Details

```
from ovirtsdk.api import API
from ovirtsdk.xml import params

try:
    API = API(url="https://HOST",
              username="USER",
              password="PASS",
              ca_file="ca.crt")
hostName = "HOSTNAME"
    host = API.hosts.get(hostName)
    for nic in host.nics.list():
        print "NIC Name:", nic.get_name()
        print "IPAddress:", nic.get_ip().get_address()
        print "MAC Address:", nic.get_mac().get_address()
        print "netStatus:", nic.get_status().get_state()
```

```
    API.disconnect()
except Exception as ex:
    print "Unexpected error: %s" % ex
```

The **nic.get_status().get_state()** returns the network status as either **up** or **down**.

Executing this code on a Python engine will show the network details:

```
NIC Name: eth0
IPAddress: 10.70.37.194
MAC Address: 00:1a:4a:46:24:9e
netStatus: up
```

You can also set values for a few configurations like IP, netmask, gateway bonding, enabling or disabling the bridge using the following methods:

- **nic.get_ip().set_address()**
- **nic.get_ip()set_netmask()**
- **nic.get_ip().set_gateway()**
- **nic.set_bridged()**

Report a bug

16.5. Adding a Resource to a Collection

The **add** method adds a resource to a collection. The resource to be added is created based on the parameters provided. Parameters are provided to the **add** method using an instance of an object from the *ovirtsdk.xml.params* module. Selecting the specific class from the module to use varies based on the type of resource being created.

Example 16.8. Adding a Resource to a Collection

In this example, a brick is added to the **GlusterBricks** collection.

```
    br_param = params.GlusterBrick(
        brick_dir='/exports/music_2',
        server_id='0f5a7016-bb36-4f2b-a226-3b16c3eeca0c')

    bricksParam = params.GlusterBricks()
    bricksParam.add_brick(br_param)
```

While the brick created in this example is not yet ready to run, it illustrates the process for creating any Red Hat Storage resource:

Procedure 16.1. Creating a Red Hat Storage Resource

1. Create an instance of the parameter object for the type of resource being created.
2. Identify the collection to which the resource will be added.

3. Call the **add** method of the collection passing the parameter object as a parameter.

Report a bug

16.6. Updating a Resource in a Collection

To update a resource, you must retrieve it from the collection it resides in, modify the desired parameters, and then call the **update** method for the resource to save the changes. Parameter modification is performed by using the **set_*** methods on the retrieved resource:

Example 16.9. Updating a Network Address

```
def setHostIp(hostName, hostIP):
    host = API.hosts.get(hostName)
    host.set_address(hostIP)
    host.update()
```

Report a bug

16.7. Removing a Resource from a Collection

To remove a resource, you must retrieve it from the collection that contains it and call the **delete** method on the resource.

Example 16.10. Removing a Resource from a Collection

Deleting a cluster named *DemoCluster* from the *clusters* collection:

```
cl = api.clusters.get("DemoCluster")
cl.delete()
```

Report a bug

16.8. Handling Errors

Where errors are encountered, the Software Development Kit uses exceptions to highlight them. The Software Development Kit defines exception types in addition to those defined by the Python interpreter itself. These exceptions are located in the **ovirtsdk.infrastructure.errors** module:

ConnectionError

Raised when a transport layer error has occurred.

DisconnectedError

Raised when attempting to use SDK after it was explicitly disconnected.

ImmutableError

Raised when initiating SDK while an SDK instance already exists under the same domain. Applicable to SDK versions before 3.2.

NoCertificatesError

Raised when no CA certificate is provided and **`insecure`** is **`False`**.

UnsecuredConnectionAttemptError

Raised when HTTP protocol is used while the server is running HTTPS.

MissingParametersError

Raised when using the **`get()`** method without providing **`id`** or **`name`**.

These exceptions can be caught and handled like any other Python exception:

Example 16.11. ConnectionError Exception

```
from ovirtsdk.api import API
from ovirtsdk.xml import params
from ovirtsdk.infrastructure import errors
try:
     api=API(url="https://INVALID_HOSTNAME",
username="USER",
     password="PASS",
     ca_file="ca.crt")
     except errors.ConnectionError, err:
     print "Connection failed: %s" % err
```

Report a bug

Chapter 17. Python Reference Documentation

17.1. Python Reference Documentation

Documentation generated using pydoc is available for these modules provided by the rhsc-sdk package:

- **ovirtsdk.api**
- **ovirtsdk.infrastructure.brokers**

If instead you wish to view the **pydoc** generated documentation on your local machine provide the name of the module you are interested in as an argument to the **pydoc** command:

```
$ pydoc MODULE
```

Report a bug

SDK Examples

A.1. Common Functions and Wrappers

Header

```
__test__ = False

 import config
 import states

 from time import sleep
 import logging
 import ovirtsdk.api
 from ovirtsdk.xml import params
 from ovirtsdk.infrastructure import errors
 from ovirtsdk.infrastructure import contextmanager
 from functools import wraps

 MB = 1024*1024
 GB = 1024*MB

 logging.basicConfig(filename='messages.log',level=logging.DEBUG)
 LOGGER = logging.getLogger(__name__)
 _major = int(config.OVIRT_VERSION[0])
 _minor = int(config.OVIRT_VERSION[2:])
 VERSION = params.Version(major=_major, minor=_minor)
```

disconnect

```
def disconnect():
     proxy = contextmanager.get('proxy')
     persistent_auth = contextmanager.get('persistent_auth')
     filter_header = contextmanager.get('filter')

     if proxy and persistent_auth:
         try:
             proxy.request(method='GET',
                           url='/api',
                           headers={'Filter': filter_header},
                           last=True)
         except Exception:
             pass
     contextmanager._clear(force=True)
```

getApi

```
def _getApi():
     """ Return ovirtsdk api.

     Will not create another API instance when reloading this module in
     ipython (when common.API is already defined).
     Works around problem when reloading, which would
     otherwise cause the error `ImmutableError: [ERROR]::'proxy' is
```

```
immutable.`.
     """
     try:
         return API
     except NameError:
         disconnect()
         return ovirtsdk.api.API(
                     url=config.OVIRT_URL, insecure=True,

username=config.OVIRT_USERNAME+'@'+config.OVIRT_DOMAIN,
                     password=config.OVIRT_PASSWORD)

 API = _getApi()
```

waitForState

```
def waitForState(obj, desiredStates, failStates=None,
timeout=config.TIMEOUT,
                     sampling=1, restoreState=None):
     """ Waits for oVirt object to change state using
:py:func:`time.sleep`.

     :param obj:             the oVirt object (host, VM, ...) for which
to wait
     :param desiredStates:   the desired oVirt object states, accepts
both a
                             list of states or a single state
     :param failStates:      fail if the object reaches one of these
states
     :param timeout:         (int) time in seconds to wait for desired
state
     :param sampling:        (int) how often to check state, in seconds
     :param restoreState     state, tryies to maintentce->up host, when
it is non_operational

     :raises AssertionError: when timeout is exceeded and the object
still isn't
         in the desired state or if failState was reached

     .. seealso:: :mod:`tests.states`
     """
     # 120 seconds is not enougn to wait for start
     if type(obj).__name__ == 'VM' and desiredStates == states.vm.up:
         timeout = 240

     global res

     if obj is None:
         return
     if type(desiredStates) is not list:
         desiredStates = [desiredStates]
     if type(failStates) is not list and failStates is not None:
         failStates = [failStates]
     elif failStates is None:
         failStates = []
```

```
    assert type(obj) is not str, "Bad use of 'waitForState()'"
    t = 0
    state = newState(obj)
    while state not in desiredStates and t <= timeout:
        sleep(sampling)
        t += sampling
        state = newState(obj)
        assert state not in failStates, \
        "Failed to get %s into state '%s' because it reached the fail \
        state '%s'" % (objectDescr(obj), desiredStates, state)
    if t > timeout:
        LOGGER.error("%s didn't reach one of states %s in timout \
                    of %i sec, current state is '%s'"
                    % (objectDescr(obj), str(desiredStates), timeout,
state))
    assert state in desiredStates, \
        "Failed to get %s into desired state" % objectDescr(obj)

    if type(obj).__name__ == 'VM':
        disks = obj.get_disks().list()
        if disks is not None:
            for disk in disks:
                waitForState(disk, states.disk.ok)
```

newState

```
def (obj):
    """ Obtain new state of an oVirt object.

    :param obj:         oVirt object (host, VM, storage, ...)
    :return:            (string) the new state of the object

    .. seealso:: :func:`updateObject`, :mod:`tests.states`
    """
    assert type(obj) is not str, "Bad use of 'newState()'"
    updatedObject = updateObject(obj)
    if updatedObject is None:
        LOGGER.warning("Object %s has no status" % (objectDescr(obj)))
        return None

    if type(updatedObject).__name__ == 'VMSnapshot':
        return updatedObject.snapshot_status

    status = updatedObject.status
    if status is None:
        LOGGER.warning("Object %s has no status" % (objectDescr(obj)))
        return None
    return status.state
```

objectDescr

```
""" Return ovirt object description

param obj:  ovirt object (glustervolumes, glusterbrick)
:return: (string) ...
```

```
typeName = type(obj).__name__
if 'GlusterVolume' in typeName:
  typeName = "Gluster Volume"

return "%s '%s'" % (typeName, obj.name)
```

Report a bug

A.2. Python SDK Example: Hosts

createHost

```
def createHost(clusterName, hostName, hostAddress, hostPassword):
     """ create host """
     msg = "Installing host '%s' on '%s'"
     LOGGER.info(msg % (hostAddress, clusterName))

     cluster = API.clusters.get(clusterName)
     assert api.clusters.get(
         name="c1").get_gluster_service() is True
     API.hosts.add(params.Host(
             name=hostName,
             address=hostAddress,
             cluster=cluster,
             root_password=hostPassword))
     host = API.hosts.get(hostName)
     assert host is not None

     waitForState(host, states.host.up,
             failStates = states.host.install_failed,
             timeout = config.HOST_INSTALL_TIMEOUT,
             restoreState=states.host.non_operational)
```

waitForTasks

```
 def waitForTasks(host, max_times=3, sleep_time=10):
     """
     Max 3(default) times try to deactive host, if there are running
tasks
     So try to wait about 30seconds, 3x10s(default)
     Parameters:
      * host - host to be deactivated
      * max_times - max times time try to deactive host
      * sleep_time - time to sleep between tryies
     """
     while max_times > 0:
         try:
             host.deactivate()
             break
         except errors.RequestError as er:
```

```
            max_times -= 1
            if max_times == 0:
                raise er
            sleep(sleep_time)
```

removeHost

```
 def removeHost(hostName):
     """ remove Host"""
     host = API.hosts.get(hostName)
     if host is not None:
         LOGGER.info("Deactivating host '%s'" % hostName)

         # Max 3 times try to deactive host, if there are running tasks
         # So try to wait about 30seconds, 3x10s
         waitForTasks(host)
         waitForState(host, states.host.maintenance)

         LOGGER.info("Deleting host")
         host.delete()
         assert updateObject(host) is None, "Failed to remove host"
     else:
         raise errors.RequestError("Unable to see any host")
     #dc = API.datacenters.get(config.MAIN_DC_NAME) ???
     #waitForState(dc, 'up')
```

activeDeactivateHost

```
def activeDeactiveHost(hostName):
     """ Active, deactive host """
     LOGGER.info("Activating/deactivating host '%s'" %hostName)
     host = API.hosts.get(hostName)
     waitForTasks(host)

     LOGGER.info("Waiting for maintence")
     host = API.hosts.get(hostName)
     waitForState(host, states.host.maintenance)
     host.activate()
     LOGGER.info("Waiting for 'up' state")
     waitForHostUpState(host)

     # Check DC state
     dc = API.datacenters.get(config.MAIN_DC_NAME)
     waitForState(dc, 'up')
```

checkHostStatus

```
def checkHostStatus(hostName):
     """ Check if is status up -> do UP """
     host = API.hosts.get(hostName)
     if host is None:
         LOGGER.info("Host '%s' dont exists." % hostName)
         return
     if host.status.state != states.host.up:
         LOGGER.info("Host '%s' state is '%s'" % (hostName,
```

```
host.status.state))
        if host.status.state != states.host.maintenance:
            host.deactivate()
            waitForState(host, states.host.maintenance, timeout=180)
        LOGGER.info("Activating")
        host.activate()
        #waitForState(host, states.host.up)
        waitForHostUpState(host)
```

configureHostNetwork

```
def configureHostNetwork(hostName):
    """
    Try to change network properties.
    Parameters:
     * hostName - name of host to be changed
    """
    h = getFilterHeader()
    # Deactive host - need to be before configuring network
    loginAsAdmin()
    host = API.hosts.get(hostName)
    waitForTasks(host)
    waitForState(host, states.host.maintenance)

    try:
        host = API.hosts.get(hostName)
        loginAsUser(filter_=h)
        for nic in host.nics.list():
            if nic.status.state == 'up':
                nic.set_boot_protocol("dhcp")
                nic.update()
                break
    except Exception as e:
        raise e
    else:
        pass
    finally:
        # Activate host after test
        loginAsAdmin()
        host = API.hosts.get(hostName)
        host.activate()
        waitForHostUpState(host)
        dc = API.datacenters.get(config.MAIN_DC_NAME)
        waitForState(dc, states.host.up)
        loginAsUser(filter_=h)
```

waitForHostUpState

```
maxTry = 3
 def waitForHostUpState(host):
    """
    Wait for host, when its state is up.
    Wait for 3x 240s. Could happend that host don't come up, so try
again.
    Parameters:
```

```
        * host - host that should be wait for
    """
    try:
        waitForState(host, states.host.up, timeout=240)
    except Exception as e:
        global maxTry
        maxTry -= 1
        if maxTry == 0:
            maxTry = 3
            raise e
        if host.status.state == states.host.non_operational or\
                host.status.state == states.host.unassigned:
            host.deactivate()
            waitForState(host, states.host.maintenance)
            host.activate()
            waitForHostUpState(host)
```

Report a bug

A.3. Python SDK Example: Cluster

createCluster

```
def createCluster(name, datacenterName,
                      cpu_type=config.HOST_CPU_TYPE, version=VERSION,
enable_virt=False, enable_gluster=True):
    """ Creates cluster """
    LOGGER.info("create_cluster")
    dc = API.datacenters.get(datacenterName)
    API.clusters.add(params.Cluster(
                name=name,
                cpu=params.CPU(id=cpu_type),
                data_center=dc,
                version=VERSION,
                virt_service=enable_virt,
                gluster_service=enable_gluster))
    cluster = API.clusters.get(name)
    LOGGER.info("Creating cluster '%s'" % name)
    assert cluster is not None
```

Example A.1. Building a cluster object

You can build a cluster object by passing the required parameters to the **`api.clusters.add()`** function.

```
api.clusters.add(params.Cluster(name="mytest.cluster.3",
cpu=params.CPU(id="Intel Nehalem Family"),
data_center=dc, virt_service=False, gluster_service=True))
```

removeCluster

```
def removeCluster(name):
```

```
    """ Removes cluster """
    cluster = API.clusters.get(name)
    if cluster is not None:
        cluster.delete()
        LOGGER.info("Removing cluster '%s'" % name)
        assert updateObject(cluster) is None, "Can't remove cluster"
```

Report a bug

A.4. Python SDK Example: Volumes

Create Volume

When a volume is created, the volume object needs a Red Hat Storage volume name, volume type and the bricks to be associated with the volume. Before creating a volume, you need to build a corresponding cluster object and associate that cluster with brick objects.

```
try:
    api = API(url=URL,
              username=USERNAME,
              password=PASSWORD,
              insecure=True)

    brick1 = params.GlusterBrick(
        brick_dir='/exports/music_1',
        server_id='0f5a7016-bb36-4f2b-a226-3b16c3eeca0c')

    brick2 = params.GlusterBrick(
        brick_dir='/exports/music_2',
        server_id='0f5a7016-bb36-4f2b-a226-3b16c3eeca0c')

    bricksParam = params.GlusterBricks()
    bricksParam.add_brick(brick1)
    bricksParam.add_brick(brick2)

    volume1 = params.GlusterVolume(
        name="music",
        cluster=api.clusters.get(name="Gluster"),
        volume_type="distribute",
        bricks=bricksParam)

    api.clusters.get("Gluster").glustervolumes.add(volume1)
    api.disconnect()

except Exception as e:
    print "Unexpected error: %s" % e
```

You can retrieve the server ID using the **`api.host`** collection.

glusterVolume Delete

```
def glusterVolumeDelete(clusterName, volumeName):
    try:

API.clusters.get(clusterName).glustervolumes.get(volumeName).delete()
```

```
        return True
    except:
        return False
```

glusterVolumeStart

To start a volume, you need to pass the cluster name with which the Red Hat Storage volume is associated and the corresponding volume name to the **`glusterVolumeStart()`** function.

```
def glusterVolumeStart(clusterName, volumeName):
    try:

API.clusters.get(clusterName).glustervolumes.get(volumeName).start()
        return True
    except:
        return False
```

glusterVolumeStop

To stop a volume, you need to pass the cluster name with which the Red Hat Storage volume is associated and the corresponding volume name to the **`glusterVolumeStart()`** function.

```
def glusterVolumeStop(clusterName, volumeName):
    try:

API.clusters.get(clusterName).glustervolumes.get(volumeName).stop()
        return True
    except:
        return False
```

Rebalance operations: startRebalance, stopRebalance, and rebalanceStatus

To rebalance data among the servers, you need to pass the cluster name and the volume name to the **`startRebalance`** and **`stopRebalance`** functions and the jobID to the **`rebalanceStatus`** function.

```
import ovirtsdk
from ovirtsdk.api import API
from ovirtsdk.xml import params

# This function returns a stop rebalance task instance. Using this
# we can reterive the status of the stop rebalance.
# ex. volume = stopRebalance(clusterName, volumeName)
#     print volume.status.state

def stopRebalance(clusterName, volumeName):
    volume = api.clusters.get(clusterName).glustervolumes.get(volumeName)
    try:
        return volume.stoprebalance()
    except ovirtsdk.infrastructure.errors.RequestError, e:
        print "Error"

def startRebalance(clusterName, volumeName):
    volume = api.clusters.get(clusterName).glustervolumes.get(volumeName)
    return volume.rebalance()

def rebalanceStatus(jobId):
```

```
    # jobs.get accepts job name and job id as an optional
    # parameters to provide the job details.
    return api.jobs.get(None, jobId)

clusterName="Default"   # name of the cluser
volumeName="music"      # name of the volume which belongs
                        # to the purticular cluster
try:
    api = API (url="https://HOST",
               username="USER",
               password="PASS",
               ca_file="ca.crt")

    status = startRebalance(clusterName, volumeName)
    jobId = status.job.get_id()
    # jobId return by startRebalance can be stored somewhere
    # for later use. With out the jobId it would be very
    # difficult to find out the actual status of this task.
    print jobId
    privStatus = None
    while True:
        status = rebalanceStatus(jobId)
        jobStatus = status.get_status().get_state()
        if jobStatus != privStatus:
            print "Description: ", status.get_description()
            print "Status:      ", jobStatus
        if "FINISHED" == jobStatus:
            break
        privStatus = jobStatus

    api.disconnect()
except Exception as ex:
    print "Unexpected error: %s" % ex
```

Report a bug

A.5. Python SDK Example: Bricks

add_Brick

```
brick3 = params.GlusterBrick(
        brick_dir='/exports/music_3',
        server_id='0f5a7016-bb36-4f2b-a226-3b16c3eeca0c')

    brick4 = params.GlusterBrick(
        brick_dir='/exports/music_4',
        server_id='0f5a7016-bb36-4f2b-a226-3b16c3eeca0c')

    bricksParam = params.GlusterBricks()
    bricksParam.add_brick(brick3)
    bricksParam.add_brick(brick4)
```

remove_Brick

```
try:

    for volume in api.clusters.get(
            CLUSTERNAME).glustervolumes.list():
        for brick in volume.bricks.list():
            if brick.name == BRICKNAME:
                brick.delete()

    api.disconnect()

except Exception as e:
    print "Unexpected error: %s" % e
```

Remove brick operations: startRemoveBrick and removeBrickStatus

You can start a remove brick operation and query the brick status using the code snippet below:

```
import ovirtsdk
from ovirtsdk.api import API
from ovirtsdk.xml import params

api = None
clusterName = "Default"   # name of the cluster
volumeName = "music"      # name of the volume which belongs #to the
particular cluster
brickName = "10.70.42.164:/home/music_b1"
# brick name. We can get the brick details which are associated to the
#volume using getGlusterVolumeBrickDetails(..)

def startRemoveBrick(clusterName, volumeName, brickName):
    volume = api.clusters.get(clusterName).glustervolumes.get(volumeName)
    bricks = volume.get_bricks()
    if not bricks:
        return None
    brick = bricks.get(brickName)
    if not brick:
        return None
    try:
        return brick.delete()
    except ovirtsdk.infrastructure.errors.RequestError as e:
        print e

def removeBrickStatus(jobId):
    # jobs.get accepts job name and job id as an optional
    # parameters to provide the job details.
    return api.jobs.get(None, jobId)

try:
    api = API (url="https://HOST",
          username="USER@domain",
          password="PASS",
          ca_file="ca.crt")

    status = startRemoveBrick(clusterName, volumeName, brickName)
    if status:
        jobId = status.job.get_id()
```

```
        privStatus = None
        while True:
            status = removeBrickStatus(jobId)
            jobStatus = status.get_status().get_state()
            if jobStatus != privStatus:
                print "Description: ", status.get_description()
                print "Status:      ", jobStatus
            if "FINISHED" == jobStatus:
                break
            privStatus = jobStatus
    else:
        print "Unable to retrieve the brick details"

    api.disconnect()
except Exception as ex:
    print "Unexpected error: %s" % ex
```

Note

Commit is not an action that can be performed on a brick. Commit happens as part of the delete action. If a remove brick action is in finished state, it commits the changes else it perform a force delete operation.

Important

The code snippet above forcefully remove the brick without migrating the data on the brick.

Report a bug

A.6. Python SDK Example: Permissions

getRoles

```
def getRoles():
     """ Return list of all roles """
     return [role.get_name() for role in API.roles.list()]
```

getRolePermissions

```
def getRolePermissions(roleName):
     """ Return permissions of role """
     role = API.roles.get(roleName)
     return [perm.get_name() for perm in role.get_permits().list()]
```

getSuperUserPermissions

```
def getSuperUserPermissions():
     """ Return SuperUser permissions(all possible permissions) """
     return getRolePermissions('SuperUser')
```

addRoleToUser

```
 def addRoleToUser(roleName, userName=config.USER_NAME,
domainName=config.USER_DOMAIN):
     """
     Add system permissions to user.
     Parameters:
      * roleName - role permissions to add
      * userName - name of user who will be added permissions
      * domainName - domain of user
     """
     LOGGER.info("Adding role '%s' to user '%s'" % (roleName, userName))
     user = getUser(userName, domainName)
     if user is None:
         return
     user.roles.add(API.roles.get(roleName))
     assert user.roles.get(roleName) is not None
```

removeAllRolesFromUser

```
def removeAllRolesFromUser(userName=config.USER_NAME,
domainName=config.USER_DOMAIN):
     """
     Removes all permissions from user.
     Parameters:
      * userName - name of user
      * domainName - domain of user
     """
     LOGGER.info("Removing all roles from user %s" % userName)
     user = getUser(userName, domainName)
     if user is None:
         return

     for role in user.roles.list():
         LOGGER.info("Removing " + role.get_name())
         role.delete()

     assert len(user.roles.list()) == 0, "Unable to remove roles from
user '%s'" % user.get_name()
```

removeRoleFromUser

```
def removeRoleFromUser(roleName, userName=config.USER_NAME,
domainName=config.USER_DOMAIN):
     """
     Remove role(System permissions) from user.
     Parameters:
      * roleName - name of role
      * userName - name of user
      * domainName - domain of user
     """
     LOGGER.info("Removing role %s to user %s" % (roleName, userName))
     user = getUser(userName, domainName)
     if user is None:
         return
```

```
    role = user.roles.get(roleName)
    role.delete()

    role = user.roles.get(roleName)
    assert role is None, "Unable to remove role '%s'" % roleName
```

givePermissionsToGroup

```
 def givePermissionsToGroup(templateName, roleName='UserTemplateBasedVm',
group="Everyone"):
    """
    Give permission to group.
    Parameters:
     * templateName - name of template to add group perms
     * roleName     - name of role which perms to be added
     * group        - On which group should be perms added
    """
    template = getObjectByName(API.templates, templateName)
    r = API.roles.get(roleName)

    g = API.groups.get(group)
    g.permissions.add(params.Permission(role=r, template=template))
    LOGGER.info("Adding permissions on template '%s' role '%s' for group
'%s'.",
                template.get_name(), roleName, group)
```

givePermissionToObject

```
 def givePermissionToObject(rhsc_object, roleName,
userName=config.USER_NAME,
                               domainName=config.USER_DOMAIN,
user_object=None,
                               role_object=None):
    """
    Add role permission to user on object.
    Parameters:
     * rhsc_object - object to add role permissions on
     * roleName     - Role permissions to be added
     * userName     - user who should be added permissions
     * domainName   - domain of user
     * user_object  - temporaly, because uf bug 869334
     * role_object  - temporaly, because uf bug 869334
    """
    # FIXME: rhsc_object can be one of:
    # [API.clusters, API.datacenters, API.disks, API.groups, API.hosts,
    #  API.storagedomains, API.templates, API.vms, API.vmpools]

    try:
        user = getUser(userName, domainName)
        if user is None:
            return
    except errors.RequestError as e:
        # User cant access /users url. Bug 869334. Workaround
        user = user_object
```

```
    try:
        role = API.roles.get(roleName)
    except errors.RequestError as e:
        # User cant access /roles url. Bug 869334. Workaround
        role = role_object

    if rhsc_object is None or user is None or role is None:
        LOGGER.warning("Unable to add permissions on 'None' object")
        returnremoving the first digit from a line

    permissionParam = params.Permission(user=user, role=role)
    try:
        rhsc_object.permissions.add(permissionParam)
    except AttributeError as e:
        # Bz 869334 - after BZ ok, could be removed
        pass

    msg = "Added permission on '%s' with role '%s' for user '%s'"
    LOGGER.info(msg % (type(rhsc_object).__name__, roleName,
user.get_name()))
```

givePermissionToCluster

```
 def removeAllPermissionFromCluster(clusterName):
    cluster = getObjectByName(API.clusters, clusterName)
    removeAllPermissionFromObject(cluster)
```

removeAllPermissionFromObject

```
 def removeAllPermissionFromObject(rhsc_object):
    """
    Removes all permissions from object
    Parameters:
     * rhsc_object - object from which permissions should be removed
    """
    LOGGER.info("Removing all permissions from object '%s'" %
type(rhsc_object).__name__)
    if rhsc_object is None:
        LOGGER.info("Tying to remove perms from object that dont
exists")
        return

    permissions = rhsc_object.permissions.list()
    for perm in permissions:
        perm.delete()
```

removeAllPermissionFromCluster

```
 def removeAllPermissionFromCluster(clusterName):
    cluster = getObjectByName(API.clusters, clusterName)
    removeAllPermissionFromObject(cluster)
```

Report a bug

A.7. Python SDK Example: Users

A.7. Python SDK Example: Users

getFilterHeader

```
def getFilterHeader():
     """ Has user admin role or user role? """
     return contextmanager.get('filter')
```

loginAsUser

```
def loginAsUser(userName=config.USER_NAME,
                domain=config.USER_DOMAIN,
                password=config.USER_PASSWORD,
                filter_=True):
     LOGGER.info("Login as %s" % userName)
     global API
     API.disconnect()
     API = ovirtsdk.api.API(url=config.OVIRT_URL, insecure=True,
                    username=userName+'@'+domain,
                    password=password, filter=filter_)
```

loginAsAdmin

```
 def loginAsAdmin():
     loginAsUser(config.OVIRT_USERNAME,
                 config.OVIRT_DOMAIN,
                 config.OVIRT_PASSWORD,
                 filter_=False)
```

editObject

```
def editObject(rhsc_object, name, newName=None, description=None,
append=False):
     """
     Edit object property.
     Parameters:
      * rhsc_object - object to be edited
      * name - name of object
      * newName - new name of object
      * description - description to be updated
      * append - True if append to old description else create new
     """
     obj = rhsc_object.get(name)

     old_name = obj.get_name()
     old_desc = obj.get_description()
     if newName is not None:removing the first digit from a line
         LOGGER.info("Updating name from '%s' to '%s'" %(name, newName))
         obj.set_name(newName)
         obj.update()
         obj = rhsc_object.get(name)
         assert old_name != obj.get_name(), "Failed to update object
name"

     if description is not None:
```

```
        LOGGER.info("Updating desc from '%s' to '%s'" %(name,
description))
        if old_desc is None:
            old_desc = ""
        obj.set_description(old_desc + description if append else
description)
        obj.update()
        obj = rhsc_object.get(name)
        assert old_desc != obj.get_description(), "Failed to update
object description"
```

hasUserPermissions

```
def hasUserPermissions(obj, role, user=config.USER_NAME,
domain=config.USER_DOMAIN):
    """
    Tests if user have role permssions on rhsc_object
    Parameters:
     * obj    - object which we wanna tests
     * role   - role we wanna test
     * user   - user we wanna test
     * domain - domain where user belongs
    """
    perms = obj.permissions.list()
    for perm in perms:
        role_name =
API.roles.get(id=perm.get_role().get_id()).get_name()
        user_name =
API.users.get(id=perm.get_user().get_id()).get_user_name()
        if user_name + '@' + domain == user and role_name == role:
            return True
    return False
```

hasPermissions

```
def hasPermissions(role):
    """ Get a list of permissions the user role has.

    :param role:    (string) oVirt user role
    :return:        (list of strings) permissions the role should have
    """
    return getRolePermissions(role)

 # If bz plugin is not enabled, use this
 def bz(*ids):
    def decorator(func):
        return func
    return decorator

 # If tcms plugin is not enabled, use this
 def tcms(*ids):
    def decorator(func):
        return func
    return decorator
```

Report a bug

A.8. Python SDK Example: Hooks

Hook operations: **getGlusterHook**, **enableHook**, **disableHook**, **resolveHook**,**glusterHookDetails**, and **glusterHookList**

```
import pprint
import ovirtsdk
from ovirtsdk.api import API
from ovirtsdk.xml import params

api = None
clusterName = "Default"   # name of the cluser

def getGlusterHook(clusterName, hookName):
     for hook in api.clusters.get(clusterName).glusterhooks.list():
          if hookName == hook.name:
               return hook

def enableHook(clusterName, hookName):
     """ Returns enable hook status """
     hook = getGlusterHook(clusterName, hookName)
     if hook:
          return hook.enable()
     return None

def disableHook(clusterName, hookName):
     """ Returns disable hook status """
     hook = getGlusterHook(clusterName, hookName)
     if hook:
          return hook.disable()
     return None

def resolveHook(clusterName, hookName):
     """ Returns resolve hook status """
     hook = getGlusterHook(clusterName, hookName)
     if hook:
          return hook.resolve()
     return None

def glusterHookDetails(clusterName):
     # Returns hook content """
     hooks=[]
     for hook in api.clusters.get(clusterName).glusterhooks.list():
        hooks.append({"hookName": hook.name,
                      "glusterCommand": hook.get_gluster_command(),
                      "level": hook.get_stage(),#indicates whether this
is a post gluster command hook or pre gluster command hook.
                      "state": hook.get_status().state,
                      "md5sum": hook.get_checksum(),
                      "content": hook.get_content()})
     return hooks

def glusterHookList(clusterName):
```

```
     # Returns list of hook names """
     hookList=[]
     for hook in api.clusters.get(clusterName).glusterhooks.list():
        hookList.append(hook.name)
     return hookList

try:
     api = API (url="https://10.70.43.95",
                username="admin@internal",
                password="redhat",
                insecure=True)
               #ca_file="ca.crt")
     hookList = glusterHookList(clusterName)
     print hookList
     print disableHook(clusterName, hookList[-1])
     pprint.pprint(glusterHookDetails(clusterName))
     print enableHook(clusterName, hookList[-1])
     api.disconnect()
except Exception as ex:
     print "Unexpected error: %s" % ex
```

Note

The status of enable hook, disable hook, and resolve hook can be retrieved using **`get_status().state`**. And it will raise an ovirtsdk.infrastructure.errors.RequestError when there is a failure.

Report a bug

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 OS and Solaris. Most Linux distributions include **cURL** as a package.

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/).

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 Storage Console host within the API entry point path (**/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 username and password of the Red Hat Storage Console 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. See Chapter 2, *Authentication and Security* for more information on obtaining a certificate.

Example: **--cacert ~/Certificates/rhsc.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 "<bricks><brick><server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id><brick_dir>/export/data/brick3</brick_dir></brick></bricks>"**

Examples

The following examples show how to adapt REST requests to **cURL** command syntax:

Example B.1. GET request

The following **GET** request lists the clusters in the **cluster** collection. Note that a **GET** request does not contain a body.

```
GET /api/clusters HTTP/1.1
Accept: application/xml
```

Adapt the method (**GET**), header (**Accept: application/xml**) and URI (**https://*[RHSC Host]*/api/clusters**) into the following **cURL** command:

```
$ curl -X GET -H "Accept: application/xml" -u [USER:PASS] --cacert
[CERT] https://[RHSC Host]/api/clusters
```

An XML representation of the **clusters** collection displays.

Example B.2. POST request

The following **POST** request creates a volume in the **server** collection. Note that a **POST** request requires a body.

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

<gluster_volume>
  <name>data</name>
  <volume_type>DISTRIBUTE</volume_type>
  <bricks>
    <brick>
      <server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
      <brick_dir>/export/data/brick1</brick_dir>
    </brick>
    <brick>
      <server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
      <brick_dir>/export/data/brick2</brick_dir>
    </brick>
  </bricks>
</gluster_volume>
```

Adapt the method (**POST**), headers (**Accept: application/xml** and **Content-type: application/xml**), URI (**https://*[RHSC Host]*/api/clusters/*99408929-82cf-4dc7-a532-9d998063fa95*/glustervolumes**) 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] https://[RHSC
HOST]/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95/glustervolumes
-d "<gluster_volume><name>data</name>
<volume_type>DISTRIBUTE</volume_type><bricks><brick>
<server_id>fcb46b88-f32e-11e1-918a-0050568c4349</server_id>
<brick_dir>/export/data/brick1</brick_dir></brick><brick>
<server_id>de173e6a-fb05-11e1-a2fc-0050568c4349</server_id>
<brick_dir>/export/data/brick2</brick_dir></brick></bricks>
</gluster_volume>"
```

The REST API creates a new volume and displays an XML representation of the resource.

Example B.3. PUT request

The following **PUT** request updates the cluster. Note that a **PUT** request requires a body.

```
PUT /api/clusters/99408929-82cf-4dc7-a532-9d998063fa95 HTTP/1.1
Accept: application/xml
Content-type: application/xml

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

Adapt the method (**PUT**), headers (**Accept: application/xml** and **Content-type: application/xml**), URI (**https://*[RHSC Host]*/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95**) 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 "<cluster>
<description>Cluster 1</description></cluster>" https://[RHSC
Host]/api/clusters/99408929-82cf-4dc7-a532-9d998063fa95
```

The REST API updates the cluster with a new description.

Example B.4. DELETE request

The following **DELETE** request removes a cluster resource.

```
DELETE /api/clusters/082c794b-771f-452f-83c9-b2b5a19c0399 HTTP/1.1
```

Adapt the method (**DELETE**) and URI (**https://*[RHSC Host]*/api/clusters/082c794b-771f-452f-83c9-b2b5a19c0399**) into the following **cURL** command:

```
$ curl -X DELETE -u [USER:PASS] --cacert [CERT] https://[RHSC
Host]/api/clusters/082c794b-771f-452f-83c9-b2b5a19c0399
```

> The REST API removes the cluster. Note the **`Accept: application/xml`** request header is optional due to the empty result of **`DELETE`** requests.

cURL Library (libcurl)

In addition the the standard command line tools, **cURL** also features **libcurl**, a library for programming language integration. For more information on supported programming languages and integration methods, see the **libcurl** website (http://curl.haxx.se/libcurl/).

Report a bug

Event Codes

C.1. Event Codes

Table C.1. Event Codes

Code	Description
0	UNASSIGNED
1	VDC_START
2	VDC_STOP
12	VDS_FAILURE
13	VDS_DETECTED
14	VDS_RECOVER
15	VDS_MAINTENANCE
16	VDS_ACTIVATE
17	VDS_MAINTENANCE_FAILED
18	VDS_ACTIVATE
19	VDS_RECOVER_FAILED
123	VDS_SLOW_STORAGE_RESPONSE_TIME
493	VDS_ALREADY_IN_REQUESTED_STATUS
494	VDS_MANUAL_FENCE_STATUS
495	VDS_MANUAL_FENCE_STATUS_FAILED
530	VDS_MANUAL_FENCE_FAILED_CALL_FENCE_SPM
531	VDS_LOW_MEM
532	VDS_HIGH_MEM_USE
533	VDS_HIGH_NETWORK_USE
534	VDS_HIGH_CPU_USE
535	VDS_HIGH_SWAP_USE
536	VDS_LOW_SWAP
496	VDS_FENCE_STATUS
497	VDS_FENCE_STATUS_FAILED
498	VDS_APPROVE
499	VDS_APPROVE_FAILED
500	VDS_FAILED_TO_RUN_VMS
504	VDS_INSTALL
505	VDS_INSTALL_FAILED
506	VDS_INITIATED_RUN_VM
537	VDS_INITIATED_RUN_VM_AS_STATELESS
507	VDS_INITIATED_RUN_VM_FAILED
509	VDS_INSTALL_IN_PROGRESS
510	VDS_INSTALL_IN_PROGRESS_WARNING
511	VDS_INSTALL_IN_PROGRESS_ERROR
513	VDS_RECOVER_FAILED_VMS_UNKNOWN
514	VDS_INITIALIZING
515	VDS_CPU_LOWER_THAN_CLUSTER
516	VDS_CPU_RETRIEVE_FAILED
517	VDS_FAILED_TO_GET_HOST_HARDWARE_INFO

Code	Description
533	VDS_STORAGE_CONNECTION_FAILED_BUT_LAST_VDS
535	VDS_STORAGES_CONNECTION_FAILED
534	VDS_STORAGE_VDS_STATS_FAILED
517	VDS_SET_NONOPERATIONAL
518	VDS_SET_NONOPERATIONAL_FAILED
519	VDS_SET_NONOPERATIONAL_NETWORK
603	VDS_SET_NONOPERATIONAL_IFACE_DOWN
522	VDS_SET_NONOPERATIONAL_DOMAIN
523	VDS_SET_NONOPERATIONAL_DOMAIN_FAILED
524	VDS_DOMAIN_DELAY_INTERVAL
23	VDS_LOW_DISK_SPACE
24	VDS_LOW_DISK_SPACE_ERROR
600	USER_VDS_MAINTENANCE
601	CPU_FLAGS_NX_IS_MISSING
602	USER_VDS_MAINTENANCE_MIGRATION_FAILED
121	SYSTEM_VDS_RESTART
122	SYSTEM_FAILED_VDS_RESTART
604	VDS_TIME_DRIFT_ALERT
605	PROXY_HOST_SELECTION
606	HOST_REFRESHED_CAPABILITIES
607	HOST_REFRESH_CAPABILITIES_FAILED
22	IRS_FAILURE
26	IRS_DISK_SPACE_LOW
201	IRS_DISK_SPACE_LOW_ERROR
204	IRS_HOSTED_ON_VDS
30	USER_VDC_LOGIN
114	USER_VDC_LOGIN_FAILED
31	USER_VDC_LOGOUT
815	USER_VDC_LOGOUT_FAILED
150	USER_INITIATED_RUN_VM
156	USER_INITIATED_RUN_VM_AND_PAUSE
153	USER_STARTED_VM
151	USER_INITIATED_RUN_VM_FAILED
32	USER_RUN_VM
538	USER_RUN_VM_AS_STATELESS
54	USER_FAILED_RUN_VM
70	USER_RUN_VM_AS_STATELESS_FINISHED_FAILURE
171	USER_RUN_VM_AS_STATELESS_WITH_DISKS_NOT_ALLOWING_SNAPSHOT
1001	USER_RUN_VM_FAILURE_STATELESS_SNAPSHOT_LEFT
152	USER_RUN_VM_ON_NON_DEFAULT_VDS
33	USER_STOP_VM
111	USER_STOP_SUSPENDED_VM
112	USER_STOP_SUSPENDED_VM_FAILED

Code	Description
56	USER_FAILED_STOP_VM
34	USER_ADD_VM
37	USER_ADD_VM_STARTED
53	USER_ADD_VM_FINISHED_SUCCESS
60	USER_ADD_VM_FINISHED_FAILURE
57	USER_FAILED_ADD_VM
35	USER_UPDATE_VM
58	USER_FAILED_UPDATE_VM
250	USER_UPDATE_VM_CLUSTER_DEFAULT_HOST_CLEARED
113	USER_REMOVE_VM_FINISHED
172	USER_REMOVE_VM_FINISHED_WITH_ILLEGAL_DISKS
149	USER_ADD
59	USER_FAILED_REMOVE_VM
38	USER_CHANGE_DISK_VM
102	USER_FAILED_CHANGE_DISK_VM
72	USER_CHANGE_FLOPPY_VM
75	USER_FAILED_CHANGE_FLOPPY_VM
39	USER_PAUSE_VM
55	USER_FAILED_PAUSE_VM
501	USER_SUSPEND_VM
512	USER_SUSPEND_VM_FINISH_SUCCESS
521	USER_SUSPEND_VM_FINISH_FAILURE
532	USER_SUSPEND_VM_FINISH_FAILURE_WILL_TRY_AGAIN
502	USER_FAILED_SUSPEND_VM
503	USER_SUSPEND_VM_OK
40	USER_RESUME_VM
103	USER_FAILED_RESUME_VM
73	USER_INITIATED_SHUTDOWN_VM
74	USER_FAILED_SHUTDOWN_VM
76	USER_STOPPED_VM_INSTEAD_OF_SHUTDOWN
77	USER_FAILED_STOPPING_VM_INSTEAD_OF_SHUTDOWN
78	USER_ADD_DISK_TO_VM
97	USER_ADD_DISK_TO_VM_FINISHED_SUCCESS
98	USER_ADD_DISK_TO_VM_FINISHED_FAILURE
79	USER_FAILED_ADD_DISK_TO_VM
80	USER_REMOVE_DISK_FROM_VM
81	USER_FAILED_REMOVE_DISK_FROM_VM
91	USER_MOVED_VM
92	USER_MOVED_VM_FINISHED_SUCCESS
83	USER_MOVED_VM_FINISHED_FAILURE
84	USER_FAILED_MOVE_VM
93	USER_MOVED_TEMPLATE

Code	Description
94	USER_MOVED_TEMPLATE_FINISHED_SUCCESS
85	USER_MOVED_TEMPLATE_FINISHED_FAILURE
86	USER_FAILED_MOVE_TEMPLATE
95	USER_COPIED_TEMPLATE
96	USER_COPIED_TEMPLATE_FINISHED_SUCCESS
87	USER_COPIED_TEMPLATE_FINISHED_FAILURE
88	USER_FAILED_COPY_TEMPLATE
89	USER_UPDATE_VM_DISK
2000	USER_FAILED_UPDATE_VM_DISK
2001	USER_HOTPLUG_DISK
2002	USER_FAILED_HOTPLUG_DISK
2003	USER_HOTUNPLUG_DISK
2004	USER_FAILED_HOTUNPLUG_DISK
2005	USER_COPIED_TEMPLATE_DISK
2006	USER_COPIED_TEMPLATE_DISK_FINISHED_SUCCESS
2007	USER_COPIED_TEMPLATE_DISK_FINISHED_FAILURE
2008	USER_MOVED_VM_DISK
2009	USER_FAILED_MOVED_VM_DISK
2010	USER_MOVED_VM_DISK_FINISHED_SUCCESS
2011	USER_MOVED_VM_DISK_FINISHED_FAILURE
2014	USER_FINISHED_REMOVE_DISK
2015	USER_FINISHED_FAILED_REMOVE_DISK
2016	USER_ATTACH_DISK_TO_VM
2017	USER_FAILED_ATTACH_DISK_TO_VM
2018	USER_DETACH_DISK_FROM_VM
2019	USER_FAILED_DETACH_DISK_FROM_VM
2020	USER_ADD_DISK
2021	USER_ADD_DISK_FINISHED_SUCCESS
2022	USER_ADD_DISK_FINISHED_FAILURE
2023	USER_FAILED_ADD_DISK
2024	USER_RUN_UNLOCK_ENTITY_SCRIPT
2025	USER_MOVE_IMAGE_GROUP_FAILED_TO_DELETE_SRC_IMAGE
2026	USER_MOVE_IMAGE_GROUP_FAILED_TO_DELETE_DST_IMAGE
3000	USER_ADD_QUOTA
3001, 3002	USER_FAILED_ADD_QUOTA
3003	USER_FAILED_UPDATE_QUOTA
3004	USER_DELETE_QUOTA
3005	USER_FAILED_DELETE_QUOTA
3006	USER_EXCEEDED_QUOTA_VDS_GROUP_GRACE_LIMIT
3007	USER_EXCEEDED_QUOTA_VDS_GROUP_LIMIT

Code	Description
3008	USER_EXCEEDED_QUOTA_VDS_GROUP_THRESHOLD
3009	USER_EXCEEDED_QUOTA_STORAGE_GRACE_LIMIT
3010	USER_EXCEEDED_QUOTA_STORAGE_LIMIT
3011	USER_EXCEEDED_QUOTA_STORAGE_THRESHOLD
3012	QUOTA_STORAGE_RESIZE_LOWER_THEN_CONSUMPTION
3013	MISSING_QUOTA_STORAGE_PARAMETERS_PERMISSIVE_MODE
3014	MISSING_QUOTA_CLUSTER_PARAMETERS_PERMISSIVE_MODE
4000	GLUSTER_VOLUME_CREATE
4001	GLUSTER_VOLUME_CREATE_FAILED
4002	GLUSTER_VOLUME_OPTION_ADDED
4003	GLUSTER_VOLUME_OPTION_SET_FAILED
4004	GLUSTER_VOLUME_START
4005	GLUSTER_VOLUME_START_FAILED
4006	GLUSTER_VOLUME_STOP
4007	GLUSTER_VOLUME_STOP_FAILED
4008	GLUSTER_VOLUME_OPTIONS_RESET
4009	GLUSTER_VOLUME_OPTIONS_RESET_FAILED
4010	GLUSTER_VOLUME_DELETE
4011	GLUSTER_VOLUME_DELETE_FAILED
4012	GLUSTER_VOLUME_REBALANCE_START
4013	GLUSTER_VOLUME_REBALANCE_START_FAILED
4014	GLUSTER_VOLUME_REMOVE_BRICKS
4015	GLUSTER_VOLUME_REMOVE_BRICKS_FAILED
4016	GLUSTER_VOLUME_REPLACE_BRICK_FAILED
4017	GLUSTER_VOLUME_REPLACE_BRICK_START
4018	GLUSTER_VOLUME_REPLACE_BRICK_START_FAILED
4019	GLUSTER_VOLUME_ADD_BRICK
4020	GLUSTER_VOLUME_ADD_BRICK_FAILED
4021	GLUSTER_SERVER_REMOVE_FAILED
4022	GLUSTER_VOLUME_PROFILE_START
4023	GLUSTER_VOLUME_PROFILE_START_FAILED
4024	GLUSTER_VOLUME_PROFILE_STOP
4025	GLUSTER_VOLUME_PROFILE_STOP_FAILED
4026	GLUSTER_VOLUME_CREATED_FROM_CLI
4027	GLUSTER_VOLUME_DELETED_FROM_CLI
4028	GLUSTER_VOLUME_OPTION_SET_FROM_CLI
4029	GLUSTER_VOLUME_OPTION_RESET_FROM_CLI
4030	GLUSTER_VOLUME_PROPERTIES_CHANGED_FROM_CLI
4031	GLUSTER_VOLUME_BRICK_ADDED_FROM_CLI

Code	Description
4032	GLUSTER_VOLUME_BRICK_REMOVED_FROM_CLI
4033	GLUSTER_SERVER_REMOVED_FROM_CLI
4034	GLUSTER_VOLUME_INFO_FAILED
4035	GLUSTER_COMMAND_FAILED
4036	GLUSTER_SERVER_ADD_FAILED
4037	GLUSTER_SERVERS_LIST_FAILED
4038	GLUSTER_SERVER_REMOVE
4039	GLUSTER_VOLUME_STARTED_FROM_CLI
4040	GLUSTER_VOLUME_STOPPED_FROM_CLI
4041	GLUSTER_VOLUME_OPTION_CHANGED_FROM_CLI
4042	GLUSTER_HOOK_ENABLE
4043	GLUSTER_HOOK_ENABLE_FAILED
4044	GLUSTER_HOOK_ENABLE_PARTIAL
4045	GLUSTER_HOOK_DISABLE
4046	GLUSTER_HOOK_DISABLE_FAILED
4047	GLUSTER_HOOK_DISABLE_PARTIAL
4048	GLUSTER_HOOK_LIST_FAILED
4049	GLUSTER_HOOK_CONFLICT_DETECTED
4050	GLUSTER_HOOK_DETECTED_NEW
4051	GLUSTER_HOOK_DETECTED_DELETE
4052	GLUSTER_VOLUME_OPTION_MODIFIED
4053	GLUSTER_HOOK_GETCONTENT_FAILED
4054	GLUSTER_SERVICES_LIST_FAILED
4055	GLUSTER_SERVICE_TYPE_ADDED_TO_CLUSTER
4056	GLUSTER_CLUSTER_SERVICE_STATUS_CHANGED
4057	GLUSTER_SERVICE_ADDED_TO_SERVER
4058	GLUSTER_SERVER_SERVICE_STATUS_CHANGED
4059	GLUSTER_HOOK_UPDATED
4060	GLUSTER_HOOK_UPDATE_FAILED
4061	GLUSTER_HOOK_ADDED
4062	GLUSTER_HOOK_ADD_FAILED
4063	GLUSTER_HOOK_REMOVED
4064	GLUSTER_HOOK_REMOVE_FAILED
4065	GLUSTER_HOOK_REFRESH
4066	GLUSTER_HOOK_REFRESH_FAILED
4067	GLUSTER_SERVICE_STARTED
4068	GLUSTER_SERVICE_START_FAILED
4069	GLUSTER_SERVICE_STOPPED
4070	GLUSTER_SERVICE_STOP_FAILED
4071	GLUSTER_SERVICES_LIST_NOT_FETCHED
4072	GLUSTER_SERVICE_RESTARTED
4073	GLUSTER_SERVICE_RESTART_FAILED
4074	GLUSTER_VOLUME_OPTIONS_RESET_ALL

Code	Description
4075	GLUSTER_HOST_UUID_NOT_FOUND
4076	GLUSTER_VOLUME_BRICK_ADDED
4077	GLUSTER_CLUSTER_SERVICE_STATUS_ADDED
41	USER_VDS_RESTART
107	USER_FAILED_VDS_RESTART
20	USER_VDS_START
118	USER_FAILED_VDS_START
21	USER_VDS_STOP
137	USER_FAILED_VDS_STOP
42	USER_ADD_VDS
104	USER_FAILED_ADD_VDS
43	USER_UPDATE_VDS
105	USER_FAILED_UPDATE_VDS
44	USER_REMOVE_VDS
106	USER_FAILED_REMOVE_VDS
45	USER_CREATE_SNAPSHOT
68	USER_CREATE_SNAPSHOT_FINISHED_SUCCESS
69	USER_CREATE_SNAPSHOT_FINISHED_FAILURE
170	USER_CREATE_LIVE_SNAPSHOT_FINISHED_FAILURE
117	USER_FAILED_CREATE_SNAPSHOT
342	USER_REMOVE_SNAPSHOT
343	USER_FAILED_REMOVE_SNAPSHOT
356	USER_REMOVE_SNAPSHOT_FINISHED_SUCCESS
357	USER_REMOVE_SNAPSHOT_FINISHED_FAILURE
46	USER_TRY_BACK_TO_SNAPSHOT
71	USER_TRY_BACK_TO_SNAPSHOT_FINISH_SUCCESS
99	USER_TRY_BACK_TO_SNAPSHOT_FINISH_FAILURE
115	USER_FAILED_TRY_BACK_TO_SNAPSHOT
47	USER_RESTORE_FROM_SNAPSHOT
1190	USER_RESTORE_FROM_SNAPSHOT_START
100	USER_RESTORE_FROM_SNAPSHOT_FINISH_SUCCESS
101	USER_RESTORE_FROM_SNAPSHOT_FINISH_FAILURE
116	USER_FAILED_RESTORE_FROM_SNAPSHOT
48	USER_ADD_VM_TEMPLATE
51	USER_ADD_VM_TEMPLATE_FINISHED_SUCCESS
52	USER_ADD_VM_TEMPLATE_FINISHED_FAILURE
108	USER_FAILED_ADD_VM_TEMPLATE

Code	Description
49	USER_UPDATE_VM_TEMPLATE
109	USER_FAILED_UPDATE_VM_TEMPLATE
50	USER_REMOVE_VM_TEMPLATE
251	USER_REMOVE_VM_TEMPLATE_FINISHED
110	USER_FAILED_REMOVE_VM_TEMPLATE
135	TEMPLATE_IMPORT
136	TEMPLATE_IMPORT_FAILED
520	USER_ATTACH_USER_TO_VM
360	USER_DETACH_USER_FROM_VM
361	USER_FAILED_DETACH_USER_FROM_VM
182	USER_FAILED_ATTACH_USER_TO_VM
325	USER_REMOVE_ADUSER
326	USER_FAILED_REMOVE_ADUSER
327	USER_FAILED_ADD_ADUSER
346	USER_PASSWORD_CHANGED
347	USER_PASSWORD_CHANGE_FAILED
348	USER_CLEAR_UNKNOWN_VMS
349	USER_FAILED_CLEAR_UNKNOWN_VMS
528	USER_EJECT_VM_DISK
529	USER_EJECT_VM_FLOPPY
61	VM_DOWN
119	VM_DOWN_ERROR
62	VM_MIGRATION_START
63	VM_MIGRATION_DONE
64	VM_MIGRATION_ABORT
65	VM_MIGRATION_FAILED
66	VM_FAILURE
67	VM_MIGRATION_START_SYSTEM_INITIATED
120	VM_MIGRATION_FAILED_FROM_TO
128	VM_MIGRATION_TRYING_RERUN
161	VM_CANCEL_MIGRATION
162	VM_CANCEL_MIGRATION_FAILED
163	VM_STATUS_RESTORED
140	VM_MIGRATION_FAILED_DURING_MOVE_TO_MAINTENANCE
142	VM_SET_TO_UNKNOWN_STATUS
143	VM_WAS_SET_DOWN_DUE_TO_HOST_REBOOT_OR_MANUAL_FENCE
131	USER_EXPORT_VM
132	USER_EXPORT_VM_FAILED
133	USER_EXPORT_TEMPLATE
134	USER_EXPORT_TEMPLATE_FAILED
124	VM_IMPORT
125	VM_IMPORT_FAILED
126	VM_NOT_RESPONDING
127	VDS_RUN_IN_NO_KVM_MODE
141	VDS_VERSION_NOT_SUPPORTED_FOR_CLUSTER

Code	Description
129	VM_CLEARED
138	VM_PAUSED_ENOSPC
139	VM_PAUSED_ERROR
144	VM_IMPORT_INFO
145	VM_PAUSED_EIO
146	VM_PAUSED_EPERM
147	VM_POWER_DOWN_FAILED
300	USER_ADD_VM_POOL
301	USER_ADD_VM_POOL_FAILED
302	USER_ADD_VM_POOL_WITH_VMS
303	USER_ADD_VM_POOL_WITH_VMS_FAILED
320	USER_ADD_VM_POOL_WITH_VMS_ADD_VDS_FAILED
304	USER_REMOVE_VM_POOL
305	USER_REMOVE_VM_POOL_FAILED
306	USER_ADD_VM_TO_POOL
307	USER_ADD_VM_TO_POOL_FAILED
308	USER_REMOVE_VM_FROM_POOL
309	USER_REMOVE_VM_FROM_POOL_FAILED
310	USER_ATTACH_USER_TO_POOL
472	USER_ATTACH_USER_TO_POOL_INTERNAL
311	USER_ATTACH_USER_TO_POOL_FAILED
473	USER_ATTACH_USER_TO_POOL_FAILED_INTERNAL
312	USER_DETACH_USER_FROM_POOL
313	USER_DETACH_USER_FROM_POOL_FAILED
314	USER_UPDATE_VM_POOL
315	USER_UPDATE_VM_POOL_FAILED
316	USER_ATTACH_USER_TO_VM_FROM_POOL
318	USER_ATTACH_USER_TO_VM_FROM_POOL_FINISHED_SUCCESS
319	USER_ATTACH_USER_TO_VM_FROM_POOL_FINISHED_FAILURE
317	USER_ATTACH_USER_TO_VM_FROM_POOL_FAILED
344	USER_UPDATE_VM_POOL_WITH_VMS
345	USER_UPDATE_VM_POOL_WITH_VMS_FAILED
358	USER_VM_POOL_MAX_SUBSEQUENT_FAILURES_REACHED
328	USER_ATTACH_USER_TO_TIME_LEASED_POOL
329	USER_ATTACH_USER_TO_TIME_LEASED_POOL_FAILED
330	USER_DETACH_USER_FROM_TIME_LEASED_POOL
331	USER_ATTACH_USER_FROM_TIME_LEASED_POOL_FAILED
332	USER_ATTACH_AD_GROUP_FROM_TIME_LEASED_POOL

Code	Description
333	USER_ATTACH_AD_GROUP_TO_TIME_LEASED_POOL_FAILED
334	USER_DETACH_AD_GROUP_FROM_TIME_LEASED_POOL
335	USER_DETACH_AD_GROUP_FROM_TIME_LEASED_POOL_FAILED
336	USER_UPDATE_USER_TO_TIME_LEASED_POOL
337	USER_UPDATE_USER_TO_TIME_LEASED_POOL_FAILED
338	USER_UPDATE_AD_GROUP_TO_TIME_LEASED_POOL
337	USER_UPDATE_AD_GROUP_TO_TIME_LEASED_POOL_FAILED
338	USER_UPDATE_AD_GROUP_TO_TIME_LEASED_POOL
339	USER_UPDATE_AD_GROUP_TO_TIME_LEASED_POOL_FAILED
350	USER_ADD_BOOKMARK
351	USER_ADD_BOOKMARK_FAILED
352	USER_UPDATE_BOOKMARK
353	USER_UPDATE_BOOKMARK_FAILED
354	USER_REMOVE_BOOKMARK
355	USER_REMOVE_BOOKMARK_FAILED
400	USER_ATTACH_VM_TO_AD_GROUP
401	USER_ATTACH_VM_TO_AD_GROUP_FAILED
402	USER_DETACH_VM_TO_AD_GROUP
403	USER_DETACH_VM_TO_AD_GROUP_FAILED
404	USER_ATTACH_VM_POOL_TO_AD_GROUP
470	USER_ATTACH_VM_POOL_TO_AD_GROUP_INTERNAL
405	USER_ATTACH_VM_POOL_TO_AD_GROUP_FAILED
471	USER_ATTACH_VM_POOL_TO_AD_GROUP_FAILED_INTERNAL
406	USER_DETACH_VM_POOL_TO_AD_GROUP
407	USER_DETACH_VM_POOL_TO_AD_GROUP_FAILED
408	USER_REMOVE_AD_GROUP
409	USER_REMOVE_AD_GROUP_FAILED
430	USER_UPDATE_TAG
431	USER_UPDATE_TAG_FAILED
432	USER_ADD_TAG
433	USER_UPDATE_TAG_FAILED
434	USER_REMOVE_TAG
435	USER_REMOVE_TAG_FAILED
436	USER_ATTACH_TAG_TO_USER
437	USER_ATTACH_TAG_TO_USER_FAILED
438	USER_ATTACH_TAG_TO_USER_GROUP

Code	Description
439	USER_ATTACH_TAG_TO_USER_GROUP_FAILED
440	USER_ATTACH_TAG_TO_VM
441	USER_ATTACH_TAG_TO_VM_FAILED
442	USER_ATTACH_TAG_TO_VDS
443	USER_ATTACH_TAG_TO_VDS_FAILED
444	USER_DETACH_VDS_FROM_TAG
445	USER_DETACH_VDS_FROM_TAG_FAILED
446	USER_DETACH_VM_FROM_TAG
447	USER_DETACH_VM_FROM_TAG_FAILED
448	USER_DETACH_USER_FROM_TAG
449	USER_DETACH_USER_FROM_TAG_FAILED
450	USER_DETACH_USER_GROUP_FROM_TAG
451	USER_DETACH_USER_GROUP_FROM_TAG_FAILED
452	USER_ATTACH_TAG_TO_USER_EXISTS
453	USER_ATTACH_TAG_TO_USER_GROUP_EXISTS
454	USER_ATTACH_TAG_TO_VM_EXISTS
455	USER_ATTACH_TAG_TO_VDS_EXISTS
555	USER_MOVE_TAG
556	USER_MOVE_TAG_FAILED
456	USER_LOGGED_IN_VM
457	USER_LOGGED_OUT_VM
458	USER_LOCKED_VM
459	USER_UNLOCKED_VM
460	USER_DETACH_USER_FROM_TIME_LEASED_POOL_INTERNAL
461	USER_DETACH_USER_FROM_TIME_LEASED_POOL_FAILED_INTERNAL
462	USER_DETACH_AD_GROUP_FROM_TIME_LEASED_POOL_INTERNAL
463	USER_DETACH_AD_GROUP_FROM_TIME_LEASED_POOL_FAILED_INTERNAL
467	UPDATE_TAGS_VM_DEFAULT_DISPLAY_TYPE
468	UPDATE_TAGS_VM_DEFAULT_DISPLAY_TYPE_FAILED
809	USER_ADD_VDS_GROUP
810	USER_ADD_VDS_GROUP_FAILED
811	USER_UPDATE_VDS_GROUP
812	USER_UPDATE_VDS_GROUP_FAILED
835	SYSTEM_UPDATE_VDS_GROUP
836	SYSTEM_UPDATE_VDS_GROUP_FAILED
813	USER_REMOVE_VDS_GROUP
814	USER_REMOVE_VDS_GROUP_FAILED
816	MAC_POOL_EMPTY
833	MAC_ADDRESS_IS_IN_USE
838	MAC_ADDRESS_IS_IN_USE_UNPLUG
817	CERTIFICATE_FILE_NOT_FOUND

Code	Description
818	RUN_VM_FAILED
837	MAC_ADDRESSES_POOL_NOT_INITIALIZED
819	VDS_REGISTER_ERROR_UPDATING_HOST
820	VDS_REGISTER_ERROR_UPDATING_HOST_ALL_TAKEN
821	VDS_REGISTER_HOST_IS_ACTIVE
822	VDS_REGISTER_ERROR_UPDATING_NAME
823	VDS_REGISTER_ERROR_UPDATING_NAMES_ALL_TAKEN
824	VDS_REGISTER_NAME_IS_ACTIVE
825	VDS_REGISTER_AUTO_APPROVE_PATTERN
826	VDS_REGISTER_FAILED
827	VDS_REGISTER_EXISTING_VDS_UPDATE_FAILED
828	VDS_REGISTER_SUCCEEDED
834	VDS_REGISTER_EMPTY_ID
829	VM_MIGRATION_ON_CONNECT_CHECK_FAILED
830	VM_MIGRATION_ON_CONNECT_CHECK_SUCCEEDED
920	NETWORK_ATTACH_NETWORK_TO_VDS
921	NETWORK_ATTACH_NETWORK_TO_VDS_FAILED
922	NETWORK_DETACH_NETWORK_FROM_VDS
923	NETWORK_DETACH_NETWORK_FROM_VDS_FAILED
924	NETWORK_ADD_BOND
925	NETWORK_ADD_BOND_FAILED
926	NETWORK_REMOVE_BOND
927	NETWORK_REMOVE_BOND_FAILED
928	NETWORK_VDS_NETWORK_MATCH_CLUSTER
929	NETWORK_VDS_NETWORK_NOT_MATCH_CLUSTER
930	NETWORK_REMOVE_VM_INTERFACE
931	NETWORK_REMOVE_VM_INTERFACE_FAILED
932	NETWORK_ADD_VM_INTERFACE
933	NETWORK_ADD_VM_INTERFACE_FAILED
934	NETWORK_UPDATE_VM_INTERFACE
935	NETWORK_UPDATE_VM_INTERFACE_FAILED
936	NETWORK_ADD_TEMPLATE_INTERFACE
937	NETWORK_ADD_TEMPLATE_INTERFACE_FAILED
938	NETWORK_REMOVE_TEMPLATE_INTERFACE
939	NETWORK_REMOVE_TEMPLATE_INTERFACE_FAILED
940	NETWORK_UPDATE_TEMPLATE_INTERFACE
941	NETWORK_UPDATE_TEMPLATE_INTERFACE_FAILED
942	NETWORK_ADD_NETWORK

Code	Description
943	NETWORK_ADD_NETWORK_FAILED
944	NETWORK_REMOVE_NETWORK
945	NETWORK_REMOVE_NETWORK_FAILED
946	NETWORK_ATTACH_NETWORK_TO_VDS_GROUP
947	NETWORK_ATTACH_NETWORK_TO_VDS_GROUP_FAILED
948	NETWORK_DETACH_NETWORK_TO_VDS_GROUP
949	NETWORK_DETACH_NETWORK_TO_VDS_GROUP_FAILED
1102	NETWORK_ACTIVATE_VM_INTERFACE_SUCCESS
1103	NETWORK_ACTIVATE_VM_INTERFACE_FAILURE
1104	NETWORK_DEACTIVATE_VM_INTERFACE_SUCCESS
1105	NETWORK_DEACTIVATE_VM_INTERFACE_FAILURE
1100	NETWORK_UPDATE_DISPLAY_TO_VDS_GROUP
1101	NETWORK_UPDATE_DISPLAY_TO_VDS_GROUP_FAILED
1102	NETWORK_UPDATE_NETWORK_TO_VDS_INTERFACE
1103	NETWORK_UPDATE_NETWORK_TO_VDS_INTERFACE_FAILED
1104	NETWORK_COMMINT_NETWORK_CHANGES
1105	NETWORK_COMMINT_NETWORK_CHANGES_FAILED
1106	NETWORK_HOST_USING_WRONG_CLUSER_VLAN
1107	NETWORK_HOST_MISSING_CLUSER_VLAN
1108	VDS_NETWORK_MTU_DIFFER_FROM_LOGICAL_NETWORK
1109	BRIDGED_NETWORK_OVER_MULTIPLE_INTERFACES
1110	VDS_NETWORKS_OUT_OF_SYNC
1112	NETWORK_UPDTAE_NETWORK_ON_CLUSTER
1113	NETWORK_UPDTAE_NETWORK_ON_CLUSTER_FAILED
1114	NETWORK_UPDATE_NETWORK
1115	NETWORK_UPDATE_NETWORK_FAILED
1116	NETWORK_UPDATE_VM_INTERFACE_LINK_UP
1117	NETWORK_UPDATE_VM_INTERFACE_LINK_DOWN
1118	INVALID_INTERFACE_FOR_MANAGEMENT_NETWORK_CONFIGURATION
1119	VLAN_ID_MISMATCH_FOR_MANAGEMENT_NETWORK_CONFIGURATION

Code	Description
1120	SETUP_NETWORK_FAILED_FOR_MANAGEMENT_NETWORK_CONFIGURATION
1121	PERSIST_NETWORK_FAILED_FOR_MANAGEMENT_NETWORK
1162	IMPORTEXPORT_STARTING_EXPORT_VM
1150	IMPORTEXPORT_EXPORT_VM
1151	IMPORTEXPORT_EXPORT_VM_FAILED
1165	IMPORTEXPORT_STARTING_IMPORT_VM
1152	IMPORTEXPORT_IMPORT_VM
1153	IMPORTEXPORT_IMPORT_VM_FAILED
1166	IMPORTEXPORT_STARTING_REMOVE_TEMPLATE
1154	IMPORTEXPORT_REMOVE_TEMPLATE
1155	IMPORTEXPORT_REMOVE_TEMPLATE_FAILED
1156	IMPORTEXPORT_EXPORT_TEMPLATE
1164	IMPORTEXPORT_STARTING_EXPORT_TEMPLATE
1157	IMPORTEXPORT_EXPORT_TEMPLATE_FAILED
1163	IMPORTEXPORT_STARTING_IMPORT_TEMPLATE
1158	IMPORTEXPORT_IMPORT_TEMPLATE
1159	IMPORTEXPORT_IMPORT_TEMPLATE_FAILED
1167	IMPORTEXPORT_STARTING_REMOVE_VM
1160	IMPORTEXPORT_REMOVE_VM
1161	IMPORTEXPORT_REMOVE_VM_FAILED
1162	IMPORTEXPORT_GET_VMS_INFO_FAILED
1168	IMPORTEXPORT_FAILED_TO_IMPORT_VM
1169	IMPORTEXPORT_FAILED_TO_IMPORT_TEMPLATE
1170	IMPORTEXPORT_IMPORT_TEMPLATE_INVALID_INTERFACES
850	USER_ADD_PERMISSION
851	USER_ADD_PERMISSION_FAILED
852	USER_REMOVE_PERMISSION
853	USER_REMOVE_PERMISSION_FAILED
854	USER_ADD_ROLE
855	USER_ADD_ROLE_FAILED
856	USER_UPDATE_ROLE
857	USER_UPDATE_ROLE_FAILED
858	USER_REMOVE_ROLE859
859	USER_REMOVE_ROLE_FAILED
860	USER_ATTACHED_ACTION_GROUP_TO_ROLE
861	USER_ATTACHED_ACTION_GROUP_TO_ROLE_FAILED
962	USER_DETACHED_ACTION_GROUP_FROM_ROLE
863	USER_DETACHED_ACTION_GROUP_FROM_ROLE_FAILED
864	USER_ADD_ROLE_WITH_ACTION_GROUP

Code	Description
865	USER_ADD_ROLE_WITH_ACTION_GROUP_FAILED
900	AD_COMPUTER_ACCOUNT_SUCCEEDED
901	AD_COMPUTER_ACCOUNT_FAILED
950	USER_ADD_STORAGE_POOL
951	USER_ADD_STORAGE_POOL_FAILED
952	USER_UPDATE_STORAGE_POOL
953	USER_UPDATE_STORAGE_POOL_FAILED
954	USER_REMOVE_STORAGE_POOL
955	USER_REMOVE_STORAGE_POOL_FAILED
918	USER_FORCE_REMOVE_STORAGE_POOL
919	USER_FORCE_REMOVE_STORAGE_POOL_FAILED
956	USER_ADD_STORAGE_DOMAIN
957	USER_ADD_STORAGE_DOMAIN_FAILED
958	USER_UPDATE_STORAGE_DOMAIN
959	USER_UPDATE_STORAGE_DOMAIN_FAILED
960	USER_REMOVE_STORAGE_DOMAIN
961	USER_REMOVE_STORAGE_DOMAIN_FAILED
962	USER_ATTACH_STORAGE_DOMAIN_TO_POOL
963	USER_ATTACH_STORAGE_DOMAIN_TO_POOL_FAILED
964	USER_DETACH_STORAGE_DOMAIN_FROM_POOL
965	USER_DETACH_STORAGE_DOMAIN_FROM_POOL_FAILED
966	USER_ACTIVATED_STORAGE_DOMAIN
967	USER_ACTIVATE_STORAGE_DOMAIN_FAILED
968	USER_DEACTIVATED_STORAGE_DOMAIN
969	USER_DEACTIVATE_STORAGE_DOMAIN_FAILED
970	SYSTEM_DEACTIVATED_STORAGE_DOMAIN
971	SYSTEM_DEACTIVATE_STORAGE_DOMAIN_FAILED
972	USER_EXTENDED_STORAGE_DOMAIN
973	USER_EXTENDED_STORAGE_DOMAIN_FAILED
974	USER_REMOVE_VG
975	USER_REMOVE_VG_FAILED
976	USER_ACTIVATE_STORAGE_POOL
977	USER_ACTIVATE_STORAGE_POOL_FAILED
978	SYSTEM_FAILED_CHANGE_STORAGE_POOL_STATUS
979	SYSTEM_CHANGE_STORAGE_POOL_STATUS_NO_HOST_FOR_SPM
980	SYSTEM_CHANGE_STORAGE_POOL_STATUS_PROBLEMATIC
981	USER_FORCE_REMOVE_STORAGE_DOMAIN
982	USER_FORCE_REMOVE_STORAGE_DOMAIN_FAILED

Code	Description
983	RECONSTRUCT_MASTER_FAILED_NO_MASTER
984	RECONSTRUCT_MASTER_DONE
985	RECONSTRUCT_MASTER_FAILED
986	SYSTEM_CHANGE_STORAGE_POOL_STATUS_PROBLEMATIC_SEARCHING_NEW_SPM
987	SYSTEM_CHANGE_STORAGE_POOL_STATUS_PROBLEMATIC_WITH_ERROR
988	USER_CONNECT_HOSTS_TO_LUN_FAILED
989	SYSTEM_CHANGE_STORAGE_POOL_STATUS_PROBLEMATIC_FROM_NON_OPERATIONAL
990	SYSTEM_MASTER_DOMAIN_NOT_IN_SYNC
991	RECOVERY_STORAGE_POOL
992	RECOVERY_STORAGE_POOL_FAILED
993	SYSTEM_CHANGE_STORAGE_POOL_STATUS_RESET_IRS
994	CONNECT_STORAGE_SERVERS_FAILED
995	CONNECT_STORAGE_POOL_FAILED
996	STORAGE_DOMAIN_ERROR
997	REFRESH_REPOSITORY_FILE_LIST_FAILED
998	REFRESH_REPOSITORY_FILE_LIST_SUCCEEDED
999	STORAGE_ALERT_VG_METADATA_CRITICALLY_FULL
1000	STORAGE_ALERT_SMALL_VG_METADATA
1002	USER_ATTACH_STORAGE_DOMAINS_TO_POOL
1003	USER_ATTACH_STORAGE_DOMAINS_TO_POOL_FAILED
1004	STORAGE_DOMAIN_TASKS_ERROR
1005	UPDATE_OVF_FOR_STORAGE_POOL_FAILED
1006	UPGRADE_STORAGE_POOL_ENCOUNTERED_PROBLEMS
1010	RELOAD_CONFIGURATIONS_SUCCESS
1101	RELOAD_CONFIGURATIONS_FAILURE
1100	USER_ACCOUNT_DISABLED_OR_LOCKED
1101	USER_ACCOUNT_PASSWORD_EXPIRED
1150	PROVIDER_ADDED
1151	PROVIDER_ADDITION_FAILED
1152	PROVIDER_UPDATED
1153	PROVIDER_UPDATE_FAILED
1154	PROVIDER_REMOVED
1155	PROVIDER_REMOVAL_FAILED
1156	PROVIDER_CERTIFICATE_CHAIN_IMPORTED
1157	PROVIDER_CERTIFICATE_CHAIN_IMPORT_FAILED
1200	ENTITY_RENAMED
9000	VDS_ALERT_FENCE_IS_NOT_CONFIGURED
9001	VDS_ALERT_FENCE_TEST_FAILED

Code	Description
9002	VDS_ALERT_FENCE_OPERATION_FAILED
9003	VDS_ALERT_FENCE_OPERATION_SKIPPED
9004	VDS_ALERT_FENCE_NO_PROXY_HOST
9005	VDS_ALERT_FENCE_STATUS_VERIFICATION_FAILED
9006	CANNOT_HIBERNATE_RUNNING_VMS_AFTER_CLUSTER_CPU_UPGRADE
9007	VDS_ALERT_SECONDARY_AGENT_USED_FOR_FENCE_OPERATION
9500	TASK_STOPPING_ASYNC_TASK
9501	TASK_CLEARING_ASYNC_TASK
9502	VDS_ACTIVATE_ASYNC
9503	VDS_ACTIVATE_FAILED_ASYNC
9504	STORAGE_ACTIVATE_ASYNC
9505	USER_ACTIVATED_STORAGE_DOMAIN_ASYNC
9506	USER_ACTIVATE_STORAGE_DOMAIN_FAILED_ASYNC
9600	**IMPORTEXPORT_IMPORT_VM_INVALID_INTERFACES**
9601	VDS_SET_NON_OPERATIONAL_VM_NETWORK_IS_BRIDGELESS
9604	EMULATED_MACHINES_INCOMPATIBLE_WITH_CLUSTER
9602	HA_VM_FAILED
9603	HA_VM_RESTART_FAILED
9701	DWH_STOPPED
9700	DWH_STARTED
9704	DWH_ERROR
9801	EXTERNAL_EVENT_NORMAL
9802	EXTERNAL_EVENT_WARNING
9803	EXTERNAL_EVENT_ERROR
9804	EXTERNAL_ALERT
9901	WATCHDOG_EVENT
10000	VDS_UNTRUSTED

Report a bug

Java Keystores

This appendix demonstrates how to import the X.509 certificate exported from the Red Hat Storage Console server (See Section 2.1, “TLS/SSL Certification” for information on certificate exports) into a new Java keystore file.

Procedure D.1. Import a certificate into a new Java keystore

This process helps a user import the **rhsc.cer** certificate from Section 2.1, “TLS/SSL Certification” into a Java keystore. This procedure requires the **keytool** management utility from the Java Development Kit (JDK) available for Linux and Windows systems.

1. Access your client machine and locate the **rhsc.cer** certificate.
2. Import the **rhsc.cer** certificate using the Java **keytool** management utility.

   ```
   keytool -importcert -v -trustcacerts -keystore restapi.jks -
   noprompt -alias rhsc -file rhsc.cer
   ```

   The **keytool** utility creates a new keystore file named **restapi.jks**.
3. **keytool** asks for the keystore password. Enter a password and **keytool** asks to verify it.
4. **keytool** adds the **rhsc.cer** certificate to the **restapi.jks** keystore. Use **keytool -list** command to view the certificate's entry in the keystore:

   ```
   keytool -list -keystore restapi.jks -storepass [password]
   ```

Important

Some versions of **keytool** parse the certificate incorrectly. If **keytool** does not recognize the certificate, convert it to a different X.509 format with the **openssl** tool:

```
openssl x509 -in rhsc.cer -out rhsc.new -outform [pem|der]
```

This creates a file called **rhsc.new** to use in place of **rhsc.cer**.

Report a bug

Certificates

E.1. Creating SSL/TLS Certificates

Summary

SSL/TLS certificates provide a layer of security for accessing your installation over HTTPS. This procedure provides instructions for creating certificates and configuring your server with them.

The following procedure requires **openssl**. To install this tool, run the following command on your server:

```
# yum install openssl
```

A Certificate Authority (CA) signs certificates to provide verifification of their validity. Web browsers contain a number of CA certificates for verifying HTTPS communication with secure websites. Some of these CAs require a fee to provide a CA pair for your server, while some are open and provide free CA pairs. This procedure describes how to generate your own Certificate Authority (CA) certificate and key for your own internal network usage.

Procedure E.1. Creating a Certificate Authority

1. Run the following command:

   ```
   # openssl req -new -x509 -keyout ca.key -out ca.crt -days 3650
   ```

   This command requests a new CA pair valid for 3650 days.

2. Enter a password to protect your CA:

   ```
   Generating a 2048 bit RSA private key
   ..........................................................................
   ..........................................................................
   ..+++
   ..........................................................................
   ..............................+++
   writing new private key to 'ca.key'
   Enter PEM pass phrase:
   Verifying - Enter PEM pass phrase:
   ```

3. Enter the following details about your organization:

   ```
   Country Name (2 letter code) [XX]:AU
   State or Province Name (full name) []:Queensland
   Locality Name (eg, city) [Default City]:Brisbane
   Organization Name (eg, company) [Default Company Ltd]:Red Hat
   Organizational Unit Name (eg, section) []:Engineering Content
   Services
   Common Name (eg, your name or your server's hostname)
   []:www.example.com
   Email Address []:dmacpher@redhat.com
   ```

   This information forms the Distinguished Name (DN) in your certificate.

Conclusion

You have created a Certificate Authority. **openssl** creates two files: **ca.key**, which is a key that administrators use to sign certificates, and **ca.crt**, which is the public CA certificate that users obtain to verify the validity of signed certificates they receive. Make sure users accessing your server have a copy of **ca.crt** so that they can import it into their client's trusted CA store.

Report a bug

E.2. Creating an SSL Certificate

Summary

The following procedure creates an SSL certificate and signs it with the CA key. SSL/TLS certificates provide a layer of security for accessing your installation over HTTPS. This procedure provides instructions for creating certificates and configuring the server with them.

The following procedure requires **openssl**. To install this tool, run the following command on your server:

```
# yum install openssl
```

Procedure E.2. Creating an SSL Certificate

1. Create a key for your server:

   ```
   # openssl genrsa -out ssl.key
   ```

   This creates an **ssl.key** file.

2. Use the key to create a signing request for your certificate:

   ```
   # openssl req -new -key ssl.key -out ssl.csr
   ```

   The signing request asks for some organization details to form the Distinguished Name (DN) in your certificate.

   ```
   Country Name (2 letter code) [XX]:IN
   State or Province Name (full name) []:Karnataka
   Locality Name (eg, city) [Default City]:Bangalore
   Organization Name (eg, company) [Default Company Ltd]:Red Hat
   Organizational Unit Name (eg, section) []:Engineering Content
   Services
   Common Name (eg, your name or your server's hostname)
   []:www.example.com
   Email Address []:psriniva@redhat.com
   ```

   This creates an **ssl.csr** signing request file.

3. Create the signed SSL certificate:

   ```
   Create the signed SSL certificate:
   ```

   openssl asks for your CA key's password.

This creates a certificate file named **ssl.crt**.

Important

The above command may result in the following error:

```
Using configuration from /etc/pki/tls/openssl.cnf
Enter pass phrase for ca.key:
/etc/pki/CA/index.txt: No such file or directory
unable to open '/etc/pki/CA/index.txt'
139883256969032:error:02001002:system library:fopen:No such file or
directory:bss_file.c:355:fopen('/etc/pki/CA/index.txt','r')
139883256969032:error:20074002:BIO routines:FILE_CTRL:system
lib:bss_file.c:357:
```

Procedure E.3. Resolving this error

1. Create the index.txt file.

```
# touch /etc/pki/CA/index.txt
```

2. Create a serial file to label the CA and all subsequent certificates.

```
# echo '1000' > /etc/pki/CA/serial
```

You will only need to do this the first time you set up the SSL certificate. Re-run the command:

```
# openssl ca -cert ca.crt -keyfile ca.key -out ssl.crt -infiles
ssl.csr
```

The **ssl.crt** and **ssl.key** form the certificate pair that your server uses to encrypt data via HTTPS.

Conclusion

You have created an SSL certificate and signed it with the CA key. **openssl** creates two files: **ca.key**, which is a key that administrators use to sign certificates, and **ca.crt**, which is the public CA certificate that users obtain to verify the validity of signed certificates they receive. Make sure users accessing your server have a copy of the **ca.crt** so that they can import it into their client's trusted CA store.

Report a bug

E.3. Configuring HTTPS Communication

Summary

Configure HTTPS to use the SSL certificate key on your system.

Procedure E.4. Configuring HTTPS Communication

1. Edit the **/etc/httpd/conf.d/ssl.conf** file and modify the following settings:

```
SSLCertificateFile [location of your ssl.crt]
SSLCertificateKeyFile [location of your ssl.key]
```

2. After editing these settings, restart your web server:

```
# service httpd restart
```

Conclusion

You have configured HTTPS to use the SSL certificate key on your system.

Report a bug

E.4. Network Security Services (NSS) Database

Network Security Services (NSS) is a set of open-source cryptography libraries that support SSL. Several Linux tools, such as **cURL**, use NSS to verify trusted SSL communication. This process helps a user import the **rhsc.cer** certificate into the local NSS database.

For an individual user, import the **rhsc.cer** certificate using the following command:

```
$ certutil -d sql:$HOME/.pki/nssdb -A -t TC -n "Red Hat Storage Console"
-i rhsc.cer
```

For all users on a system, import the **rhsc.cer** certificate as root using the following command:

```
# certutil -d sql:/etc/pki/nssdb -A -t TC -n "Red Hat Storage Console" -i
rhsc.cer
```

Report a bug

E.5. Java Keystores

This appendix demonstrates how to import the X.509 certificate exported from the Red Hat Storage Console into a new Java keystore file.

This procedure helps a user import the rhsc.cer certificate into a Java keystore. This procedure requires the **keytool** management utility from the Java Development Kit (JDK) available for Linux and Windows systems.

Procedure E.5. Import a Certificate into a New Java Keystore

1. Access your client machine and locate the **rhsc.cer** certificate.
2. Import the **rhsc.cer** certificate using the Java **keytool** management utility.

```
keytool -importcert -v -trustcacerts -keystore restapi.jks -
noprompt -alias rhsc -file rhsc.cer
```

The **keytool** utility creates a new keystore file named **restapi.jks**.

3. **keytool** asks for the **keystore** password. Enter a password and **keytool** asks to verify it.
4. **keytool** adds the **rhsc.cer** certificate to the restapi.jks keystore. Use **keytool -list** command to view the certificate's entry in the keystore:

```
keytool -list -keystore restapi.jks -storepass [password]
```

Important

Some versions of **keytool** parse the certificate incorrectly. If **keytool** does not recognize the certificate, convert it to a different X.509 format with the **openssl** tool:

```
openssl x509 -in rhsc.cer -out rhsc.new -outform [pem|der]
```

Report a bug

Revision History

Revision 3-10	Mon Sep 22 2014	Divya Muntimadugu
Version for 3.0 GA release.		