Skytap Cloud API

Documentation Version 1.14
April 2013

REVISED: 3/22/2013
WWW.SKYTAP.COM

SKYTAP CLOUD API OVERVIEW

The Skytap Cloud API is provided as a Web Services API. This provides customers with flexibility in their integration strategy. The API follows the REST Architectural Style.

The following sections provide an overview of REST and how it applies to Skytap Cloud. The rest of the document will provide the specifications for the representation of the various resources.

The Skytap Cloud API supports automation of a few core use cases, including:

- Creating and editing the parameters of configurations, templates, and their component parts;
- Starting, suspending, or shutting down one or more virtual machines;
- Providing access to virtual machines through networks and sets of URIs
- Identifying the status and usage of machines and resources.

AN INTRODUCTION TO REST

The REST Architectural Style is based on two key principles:

- Applications are modeled as **collections of elements** with simple interactions.
- The basic **infrastructure of HTTP** is used as simply and systematically as possible.

The elements of the domain are called **resources**, which are referenced through URIs. Operations on those resources are defined by the HTTP specification against those URIs, which supports simple and regular implementations.

SOME REST CONVENTIONS

In REST, references to individual resources are expressed as HTTP-styled URIs. For instance, a reference to a user (also known as a “user resource”) might have the URI:

```
https://cloud.skytap.com/users/1
```

In this example, the number “1” at the end is the **user-id**, and the whole URI is a **user-reference**.

Clients of the API can use this user-reference to retrieve the **representation** of the user resource. A representation is a collection of attributes that compose the requested resource. The representation returned by the server will display in one of the content types (e.g. XML, JSON) supported by the server. The client can edit the user resource's attributes by submitting a modified version of the representation back to the server.

By convention, different HTTP methods are mapped to different kinds of operations available to the client. The four HTTP methods you'll use are GET, PUT, POST, and DELETE. Their use is explained in the table below:

Method	Typical Use	Example(s)	Expected Result
GET	Retrieve representation of a resource or collection of resources	GET /users/ GET/users/7	Representation of a collection of users Representation of user with ID 7
PUT	Edit properties of a resource	PUT/users/7	Representation of users with updated attributes
POST	Add to a collection of resources	POST /users [with new representation body]	Representation of new user including new resource URI
DELETE	Delete a resource	DELETE users/7	User resource is deleted.

For more information about how REST operates, see this guide.

CONTENT TYPES

The Skytap API represents resources with one of two content types: XML or JSON. The HTTP **Accept** header is the supported approach to specifying the content types of HTTP responses, so requests to the Skytap API should include the HTTP header:

```
Accept: application/xml
```

OR

```
Accept: application/json
```

When PUTing or POSTing request bodies containing data, you should also specify the content type of the request body by including the HTTP header:

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

OR

```
Content-type: application/json
```

In this document, example responses are provided in XML format. Many of the resource URIs in the Skytap API also have an HTML representation that seen using a browser. For example, the representation available to a REST client at `https://cloud.skytap.com/configurations/`corresponds to the configuration seen when visiting that same URI with a browser.

REPRESENTATION OF RESOURCES

The representation of a resource consists of a number of named "fields" that represent attributes of a resource. This document includes tables listing the names of fields for the different types of resources.

Suppose we wish to interact with a hypothetical resource of type "widget". The path of the URI to the collection of widget will include its plural form: `https://cloud.skytap.com/widgets/`. The resource URI of a widget with id 3 would be: `https://cloud.skytap.com/widgets/3`.

Supposing widget resources contain a name field and a list of objects with the type "doodad" that logically belong to it, the fields might be specified so:

Field Name	Type	Access	Description
id	string	ro	string identifier
name	string	rw	human friendly name
doodads	doodad[]	rw	list of doodad representations

The XML representation of a widget would look like:

```
<xml version=”1.0” encoding=”UTF-8”?>
<widget>
       <id>3</id>
       <name>My Widget</name>
       <doodads type=”array”>
              <doodad><color>blue</color></doodad>
              <doodad><color>orange</orange></doodad>
       </doodads>
</widget>
```

Note the attribute `type="array"` used to signal array elements.

The JSON representation of the widget would look like:

```
{"id": 3, "name": "My Widget", "doodads":[{"color": "blue"}, {"color":
"orange"}]}
```

The access types are *read only (ro), read/write (rw),* and *write only (wo).* The latter access type is used for passwords that may be set but not read. The access description for array-type fields refers to whether it’s possible to edit membership of the array, not fields contained in the members of the array.

Rather than use semi-literal representations of resource URIs like this

```
https://cloud.skytap.com/configurations/3094/machines/129/interfaces/3
```

in the document, italicized placeholders are used for URI field values in canonical form:

<interface-uri>

In this case the angle brackets come from the BNF conventions for describing grammars and tokens and are not intended to suggest XML.

SUBMITTING DATA WITH PUT AND POST REQUESTS

Requests to create or modify resources are executed by submitting representations of the resources in the request body. In many cases, to edit attributes of a resource, it is not necessary to include the entire representation of the resource you are editing. Rather you can submit only the attributes that you intend to edit. For example to change the email address attribute of the user `https://cloud.skytap.com/users/7` to "alex@example.com" using a content type of `application/xml`, PUT to that resource with a request body:

```
<user>

        <email>alex@example.com</email>

</user>
```

Note that the XML representation submitted is contained within a single `<user>` element.

Likewise, using a content type of `application/json`, PUT to that resource with a request body:

```
{"email": "alex@example.com"}
```

Note that the JSON places attributes of the user at the top level of the JSON object.

For convenience, attributes can also be submitted using query string parameters in the request path in many cases. When using query string parameters it is not necessary to submit a request body. [for example to edit an the email address as in the above examples], you can use the request:

```
PUT /users/7?email=alex@example.com

Host: cloud.skytap.com

Accept: application/json
```

In this document, examples are provided using XML request bodies and query string parameters; these methods are equivalent.

Remember that some data requires special encoding depending on the type of request. For example values containing the "&" (ampersand) character should be encoded as `&amp;` in XML requests and `%26` when submitted in a query string.

SECURITY

To authorize access to Skytap Cloud resources, each API request must include a security credential encoded in an HTTP "Basic Auth" header. This is short for "Basic access authorization," the standard HTTP authentication method; for more details on this, see http://httpd.apache.org/docs/1.3/howto/auth.html#intro. By default, Skytap usernames and passwords are used with Basic Auth. If an account administrator has enabled the "Require security tokens for API requests" option under Access Policy, security tokens must be used in place of passwords. When this option is enabled, each Skytap user can find their current security token under "My Account".

This would then be encoded in a HTTP Basic Auth header, as in the example below:

```
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
```

To ensure privacy of Skytap Cloud API usage, requests and responses are sent over SSL-encrypted connections.

ERROR/STATUS REPORTING

Since operations are carried out by issuing normal HTTP requests, the HTTP standards for content encoding, error/status reporting, and security are leveraged directly. For instance, in case a request to delete the user succeeds, the HTTP status code would be "200," but if the request was for a non-existent user, the status code might be "404."

This is the list of the error status codes actually returned by the application:

HTTP Status Code	Meaning
200	Success
401	Not authorized
404	Not found
422	Invalid parameter
423	Stale Object reference
500	System error

In addition, and as appropriate, there may be a response body in a format like:

```
<errors>
  <error>…</error>
  …
</errors>
```

UNDOCUMENTED ELEMENTS

This API documentation is incomplete. We have elected to focus first on core use cases, and to not document some elements and methods that may be changed in the near future.

Therefore, as you use the API, be aware that there may be elements present in the returned structures that are not documented. There is no guarantee that they will exist in future revisions. In such cases, you should ignore them—in no case should you build dependencies on undocumented elements.

SKYTAP CLOUD RESOURCE TYPES

The following sections describe the principal resource types exposed in the API, their representations and reference URI pattern, and the key operations defined for the resource.

TEMPLATE RESOURCE

,A *template* is a specification of one or more virtual machine images, plus associated resources such as network configurations, as well as metadata such as notes and tags.

The purpose of a template is to serve as a blueprint from which any number of runnable configurations can be created. As such, a template is not directly runnable, and there are constraints on the ways a template can be modified once created. Many of the template attributes in the API are only informative, and cannot be modified unless working on an equivalent configuration. For example, the "runstate" attribute will list the state of the VMs in this template, but cannot be changed in the API.

TEMPLATE REFERENCE URI

The template reference URI is as follows:

```
<template-ref> :==  https://.../templates/<template-id>
```

TEMPLATE RESOURCE MODEL

The element name is "template," and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	string identifier
url	string	ro	URI reference for template (template-ref)
name	string	rw	human friendly name
busy	boolean	ro	Describes if template is busy
owner	URI	rw	URI of resource owner (user-ref)
public type	boolean	ro	Identifies if the template is from a public library
description	string	rw	human friendly description
vms	vm[]	ro	array of virtual machines
networks	network[]	ro	associated network resources
regions	string	ro	Region where template is housed

Note that when a list of templates is returned, only the first three fields are included.

TEMPLATE SEMANTICS

NAME

Simple text field. Searchable within Skytap Cloud UI. Limited to 255 characters.

BUSY

Many operations can make the template busy (such as changing runstates). If you attempt to perform an operation on a busy template, you will likely receive an error.

DESCRIPTION

Simple text field. Limited to 1000 characters.

OWNER

Each resource in the system has an owner, whose quota may be charged for the resource, and whose name appears in the accounting reports.

Setting the owner field of a template to a new owner changes the ownership of the template, subject to permission and quota constraints.

REGION

The region that the template is housed in

OPERATIONS ON TEMPLATE RESOURCES

DESCRIBE TEMPLATE

Request

`GET` *<template-ref>*

Response

Template element

DELETE TEMPLATE

Request

```
DELETE <template-ref>
```

Response

Status only

UPDATE ATTRIBUTE(S) OF TEMPLATE

Request

```
PUT <template-ref>?<attr-name>=<attr-value>
```

Response

Updated template element (with a new version id).

CHANGE OWNER OF TEMPLATE

This is an instance of changing an attribute. The operation will fail if the new owner does not have sufficient quota to “accept” the template.

Request

```
PUT <template-ref>?owner=<user-ref>
```

Response

Updated template element

LIST TEMPLATES

If specified, the attr/value pair acts as a simple filter on returned results.

Request

```
GET /templates[?attr=value]
```

Response

Returns a list of templates references available to user, with the following syntax:

CREATE NEW TEMPLATE

Templates are created as frozen copies of a configuration by inserting a configuration into the collection of templates.

Request

```
POST /templates?configuration_id=<id>
```

Response

Newly created template

SAVE PUBLISHED URLS IN TEMPLATE

By default, any published URLs attached to a configuration are not saved when a template is made from that configuration. To save published URLs as part of a template, you must append the line "published_sets=true" to the create command.

Request

```
POST /templates?configuration_id=<id>&publish_sets=true
```

Response

Newly created template with published URLs.

CONFIGURATION RESOURCE

Like a template, a *configuration* is a specification that describes one or more virtual machine images, associated resources, and metadata around ownership, composition and resources. Unlike a template, many more of the properties of a configuration may be modified. More importantly, a configuration can be run.

CONFIGURATION REFERENCE URI

The configuration reference URI is as follows:

```
<configuration-ref> :==  https://.../configurations/<config-id>
```

CONFIGURATION RESOURCE MODEL

The element name is "configuration," and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	identifier
url	string	ro	URI reference (*configuration-ref*)
name	string	rw	human friendly name
runstate	string	rw	runstate of configuration
owner	URI	rw	URI for owner (*user-ref*)
suspend_on_idle	integer	rw	time before idle configuration is suspended, in minutes
publish_sets	publish_set[]	ro	array of Published set resources providing anonymous access to VMs
vms	vm[]	rw	array of Virtual Machines
networks	network[]	ro	array of Virtual Networks
regions	string	ro	region in which the configuration is house

Note that when a list of configurations is returned, only the first three elements are included.

CONFIGURATION SEMANTICS

NAME

Simple text field. Searchable via Skytap Cloud UI. Limited to 255 characters.

RUNSTATE

In general, the configuration runstate reflects a summary of the states of the contained virtual machines, and it can be used to control the states of those virtual machines in parallel.

Because it can potentially represent the states of many contained virtual machines, 'runstate' is a synthesized value. The individual VMs in a configuration may have different states from one another; for instance, one could be running, one stopped, and one suspended. Since this mixture of states cannot be represented accurately by a single field, the configuration runstate is determined by looking at the range of states and choosing a single 'representative' state. The runstate results should be interpreted as follows:

- stopped—all virtual machines are stopped
- suspended—one or more virtual machines are suspended, others are stopped
- running—one or more virtual machines are running

As mentioned, setting the runstate to one of the legal target states initiates state transitions for the contained virtual machines. Note also that not all transitions are valid.

The following table describes the possible runstate transitions:

	Assigned Runstate			
Starting Runstate	**Running**	**Stopped**	**Halted**	**Suspended**
Running	No change	Graceful Shut down	Forced Shut down	Suspended
Stopped	Running Configuration	No change	No change	(invalid)
Suspended	Running Configuration	(invalid)	(invalid)	No change

- Note that the "halted" runstate does not appear as a "FROM" state. It is actually a pseudo-state; setting the runstate of a running configuration to "halted" executes a forcible transition to the "stopped" state. This is typically faster than the "stopped" transition, and is more robust in the case of errors in the guest VM.
- Also note that some other states may appear from time to time, such as "busy" or "deploying." Any state other than the states defined above should be interpreted as a sign of a configuration (or virtual machine) in transition. In particular, these are not legal "target" states.

SUSPEND_ON_IDLE

A configuration is deemed to be idle if there is no UI interaction with any of the VMs that comprise it. If desired, the configuration can be set to auto-suspend after a certain amount of idle time. For more information on Auto-Suspend, see Enabling Auto-Suspend.

OWNER

Each resource in the system has an owner, whose quota may be charged for the resource, and whose name appears in the accounting reports.

Setting the owner field of a configuration to a new owner changes the ownership of the configuration, subject to permission and quota constraints.

VMS

This is a list of the virtual machines that comprise the configuration.

The API supports adding elements to the array indirectly by means of merging templates into the configuration—see the section below—and it also supports deleting entries from the array by a DELETE request on the array member.

It is legal to have 0 entries in this array.

NETWORKS

This is an array of virtual network objects. Every configuration can have multiple networks; the number of total networks that can be created is restricted by your customer account's network quota.

REGIONS

This field lists which region the configuration is housed in, either "US-West" or "US-East." The configuration will exist in the same regions as the template it was created from.

OPERATIONS ON CONFIGURATION RESOURCES

CREATE NEW CONFIGURATION (FROM A TEMPLATE)

Create a new configuration cloned from a template. This is the only way to create new configurations.

Request

```
POST /configurations
<template_id>… </template_id>
```

Response

New configuration element

GET CONFIGURATION DESCRIPTION

Request

```
GET <configuration-ref>
```

Response

Configuration element

DELETE CONFIGURATION

Request

```
DELETE <configuration-ref>
```

Response

Status only

UPDATE ATTRIBUTE(S) OF CONFIGURATION

Request

```
PUT <configuration-ref>?<attr-name>=<attr-value>
```

Response

Updated configuration element (with a new version id)

CHANGE OWNER OF CONFIGURATION

This is an instance of changing an attribute. The operation will fail if the new owner does not have sufficient quota to "accept" the configuration.

Request

```
PUT <configuration-ref>?owner=<user-ref>
```

Response

Returns the updated configuration element

RUN/STOP/SUSPEND CONFIGURATION

This is actually asynchronous - it returns immediately, with the status changed to an intermediate statue (e.g. "busy"). You'll need to poll to discover when the desired status has been reached.

Request

```
PUT <configuration-ref>?runstate=<newstate>
```

Where <newstate> is one of "running," "stopped," "halted," or "suspended."

Response

Status (and new version)

MERGE TEMPLATE INTO CONFIGURATION

When merging template **A** into configuration **B**, this operation has the effect of copying the machines out of **A** and adding them to **B**. Interfaces of the virtual machines of **A** are adjusted to the network in **B**, and other elements of **A** (network, metadata, etc.) are discarded.

Request

```
PUT /configurations/<config-id>

<template_id>…</template_id>
```

The reference to the template to be merged in is in the HTTP BODY.

Response

Returns updated configuration element

LIST CONFIGURATIONS

Request

```
GET /configurations
```

Response

Returns a list of configurations available to user.

VIRTUAL MACHINE RESOURCE

The "virtual machine" represents an image of a single virtual computer. Virtual machines don't exist outside of configurations or templates, but a configuration or template can have multiple virtual machines.

VIRTUAL MACHINE REFERENCE URI

Virtual machines exist only within the context of a template or configuration, so references to specific virtual machines are extensions of references to those element types. The virtual machine reference URIs are as follows:

```
<vm-ref> :== <template-ref>/vms/<vm-id>

<vm-ref> :== <configuration-ref>/vms/<vm-id>
```

VIRTUAL MACHINE RESOURCE MODEL

The element name is "vm" and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	identifier
name	string	rw	human friendly name
runstate	string	rw	runstate of configuration
hardware	hardware	ro	hardware description (see below)
error	string	ro	error string for last operation
asset_id	URI	wo	reference to mounted ISO (*asset-ref*)
interfaces	interface[]	rw	array of Virtual Network Interfaces
publish_set_refs	publish_set_ref[]	ro	array of references to Published sets including this VM

VIRTUAL MACHINE SEMANTICS

NAME

Human friendly name. Limited to 100 characters.

RUNSTATE

See the discussion of configuration runstates for information about the meanings of the runstate and the legal transitions. Also, as with configuration runstates, modifying this attribute is fundamentally asynchronous. The command will start the transition from one state to the next, and will return 'busy' while the change is underway. The client will need to poll to find out when the target state has been reached.

HARDWARE

The "hardware" element is syntactically nested but logically represents a single parameter named "hardware" with defined fields:

Field Name	Type	Access	Description
cpus	Integer	rw	number of cpus (1,2,3,4,5,6,7,8)
ram	Integer	rw	memory in MB
svms	Integer	ro	number of svms used by this vm
max_cpu	Integer	ro	Maximum settable cpus for this vm
max_ram	integer	ro	Maximum settable ram for this VM

- **CPUs.** Number of CPUs allocated to this virtual machine. Valid range is 1 to 8 (depending on **max_cpu** setting). This value may be changed only when the virtual machine is stopped (e.g. not suspended or running).
- **RAM.** Amount of RAM allocated to this virtual machine. Must be between 256 and 32,768 (MB). Maximum limit depends on **max_ram** setting. This value may be changed only when the virtual machine is stopped (e.g. not suspended or running).

ERROR

In case there has been a fault or failure in performing some operation on the virtual machine, this field will be non-empty and will contain some information that will help interpret the error.

ASSET_ID

Writing an asset-ref to this field causes that asset to mount in the VM as a CD-ROM. While it's possible to specify an asset that's not an ISO image, it will fail.

Note that this field is write-only—it is always displayed as empty.

INTERFACES

An array of Virtual Network Interface resources, defined on pg. 19.

PUBLISH_SET_REFS

This is an optional array of references to Published Set resources, defined below, specifying which published sets include this VM. The published set information provided at the specified URIs contains details of the contents of the published sets and the anonymous URIs used to access them. Accessing these URIs does not require a Skytap Cloud account. Note that these references exist only when the enclosing configuration has been assigned one or more Published Set resources. If no such resources are defined then the publish_set_refs element will not be returned.

OPERATIONS ON VIRTUAL MACHINE RESOURCES

CREATE VIRTUAL MACHINE

Virtual machines are created indirectly, either by creating a configuration from a template or by merging a template into an existing configuration.

GET VIRTUAL MACHINE DESCRIPTION

Request

```
GET <vm-ref>
```

Response

Selected machine element.

UPDATE ATTRIBUTE OF VIRTUAL MACHINE

Request

```
PUT <vm-ref>?<attr-name>=<attr-value>
```

Response

Updated virtual machine element

RUN/STOP/SUSPEND VIRTUAL MACHINE

This is an instance of updating an attribute on a virtual machine.

Request

```
PUT <vm-ref>?runstate =<newstate>
```

Where <newstate> is one of “running,” “halted,” “stopped,” or “suspended.”

Response

Updated virtual machine element

MOUNT ISO ASSET ON VIRTUAL MACHINE

This is an instance of updating an attribute on a virtual machine.

Request

```
PUT <vm-ref>?asset_id=<asset-ref>
```

Response

Updated virtual machine element

DELETE VIRTUAL MACHINE

Request

```
DELETE <vm-ref>
```

Response

Status only

VIRTUAL NETWORK RESOURCE

Virtual networks are not top-level elements of the API. Rather, they are elements properly contained within a configuration, so operations on them are implicitly on the containing configuration.

VIRTUAL NETWORK REFERENCE URI

Virtual networks exist within the context of a template or configuration, so references to virtual networks are extensions of references to those elements. The virtual network reference URIs are as follows:

```
<virtual-network-ref> :== <template-ref>/networks/<network-id>
<virtual-network-ref> :== <configuration-ref>/networks/<network-id>
```

VIRTUAL NETWORK RESOURCE MODEL

The element name is “network” and the following fields are defined:

Field Name	Type	Access	Description
id	integer	ro	identifier
subnet_addr	string	rw	subnet address (X.X.X.X)
subnet_size	string	rw	size of subnet mask (1-32)
domain	string	rw	domain name
name	string	rw	User defined name of the network
network_type	string	ro	Type of network (Automatic or Manual)
tunnelable	boolean	rw	This network is visible to other networks to connect to or not
tunnels	tunnel[]	ro	List of connections between this network and other networks
primary_nameserver	string	rw	User provided primary name server for this network
secondary_nameserver	string	rw	User provided secondary name server for this network
region	string	ro	Region where the network is housed

VIRTUAL NETWORK SEMANTICS

SUBNET, DOMAIN

These values define the subnet and domain for the network to which the virtual machines in the configuration are attached. IP addresses for the virtual machines are assigned from this subnet, and standard network services (DNS resolution, CIFS share, routes to Internet) are defined appropriately for it.

These values can be changed only when *all* virtual machines in the VM are stopped (not suspended or running).

PRIMARY AND SECONDARY NAMESERVER

Skytap provisions domain name servers for 'automatic' network type. This enables VMs on the network to resolve names of other VMs on the network. Users can optionally override this automatically provisioned name server and provide their own name server for the network. These values can be used to provide up to two name servers, a primary and a secondary. If these values are set all name resolution requests are routed to these servers.

OPERATIONS ON VIRTUAL NETWORK RESOURCES

GET NETWORK INTERFACE DESCRIPTION

Request

```
GET <virtual-network-ref>
```

Response

Identified virtual network element

UPDATE VIRTUAL NETWORK

Request

```
PUT <virtual-network-ref>?<attr-name>=<attr-value>
```

Response

Updated virtual network element

VPN RESOURCE

In Skytap Cloud, a Virtual Private Network (or "VPN") refers to an IPsec network tunnel connecting a Skytap configuration to an outside server (such as a corporate intranet). The VPN resource represents a single VPN tunnel.

For more details on VPNs and their parameters, see https://cloud.skytap.com/docs/index.php/Connecting_Your_Corporate_Network_to_Skytap_Cloud.

VPN REFERENCE URI

```
<vpn-ref> :==  https://.../vpns/vpn-#
```

Note that a VPN's ID number is prefaced by "vpn-".

VPN MODEL

The element name is "vpns" and the following fields are defined:

Field Name	Type	Access	Description
Id	string	ro	identifier, in the form "vpns-#"
Name	string	rw	User-assigned name for the VPN
Status	string	ro	Status of the tunnel (inactive or active)
Enabled	boolean	rw	Whether the VPN is enabled or disabled
remote_subnets	resource	rw	resource containing remote subnets attached or excluded from the VPN; see **Remote Subnets Resource**, below.
local_peer_ip	string	rw	Public IP associated with Skytap Cloud configuration
remote_peer_ip	string	rw	Public IP associated with external server
local_subnet	boolean	rw	Range of IP addresses routed through tunnel
attached_network_count	integer	ro	Number of networks attached to VPN
connected_network_count	integer	ro	Number of networks connected to VPN
attached_networks	array	rw/ro	An array detailing attached networks (see Attached Networks resource, below)
test_results	collection	rw	Results of 4 VPN tests: Connect, Phase1, Phase2, and Ping.
phase_1_encryption_algorithm	string	rw	Name of Phase 1 encryption algorithm
phase_1_hash_algorithm	string	rw	Name of phase 1 hash algorithm
phase_1_sa_lifetime	integer	ro	Value in seconds for Phase 1 SA lifetime
phase_1_dh_group	string	rw	Name of Phase 1 Diffie-Hellman Group
phase_2_encryption_algorithm	string	rw	Name of phase 2 encryption algorithm
phase_2_authentication_algorithm	string	rw	Name of phase 2 authentication algorithm
phase_2_perfect_forward_secrecy	boolean	rw	Determines whether or not Perfect Forward Secrecy (PFS) is used.
phase_2_pfs_group	strings	rw	Shows name of PFS group if Phase 2 PFS is set to true.
phase_2_sa_lifetime	integer	rw	Value (in seconds) of Phase 2 SA Lifetime
sa_policy_level	string	rw	Security policy level, either default, use, require, or unique.
maximum_segment_size	string	rw	
dpd_enabled	boolean	rw	Determines if Dead Peer Detection is enabled.
region	string	ro	The region this VPN exists in.

OPERATIONS ON VPN RESOURCES

GET VPN DESCRIPTION

Request

```
        GET <vpn-ref>
```

Response

Representation of specified VM

CREATE NEW VPN

When POSTing a new VPN, ALL of the above fields must be entered.

ATTACHED NETWORK RESOURCE

The attached network resource contains representations of a network and information on the VPN the network is attached to. As such, the resource contains two arrays, **network** and **vpn**, that represent the two components, as well as parameters that determine the relationship between these settings. Editing or creating attached networks allows you to manage the network connections of a VPN, and the attached network resource will be contained with a VPN resource called attached_networks

ATTACHED NETWORK REFERENCE URI

```
        <attached_network-ref> :== vpns/vpn-id/attached_networks/id
```

ATTACHED NETWORK MODEL

The element name is "attached_network" and the following fields are defined:

Id	string	rw	The ID of the attached network, in the format <*network-id*>-vpn-<*vpn-id*>
connected	boolean	rw	Determines if network is connected to VPN.
network	array	ro	A read-only array containing information about the attached network, as detailed in the Network Resource section of this document.
vpn	array	ro/rw	An array containing information on the attached VPN and its status.
customer name	string	ro	The name of the customer account in which the attached network resides.
customer id	integer	ro	The customer account's ID.

ATTACHED NETWORKS SEMANTICS

NETWORK

REMOTE SUBNET RESOURCE

The remote_subnets resource contains representations of subnets associated with the VPN. These will take the form of "included" subnets (subnet ranges to which traffic from Skytap Cloud will be routed) and "excluded" subnets (subnets expressly prevented from connecting to Skytap Cloud).

REMOTE SUBNET REFERENCE URI

```
<remote_subnet-ref> :== vpns/vpn-id/subnets/remote_subnet id
```

REMOTE SUBNET MODEL

The element name is "remote_subnets" and the following fields are defined:

Field Name	Type	Access	Description
id	string	rw	The identifier of the remote subnet (for example, 22.22.0.0/16)
cidr_block	string	rw	The IP range of the subnet.
excluded	boolean	rw	If set to false, this string specifies an included subnet; if set to true, it specifies an excluded subnet.

OPERATIONS ON VPN RESOURCES

GET REMOTE SUBNET DESCRIPTION

Request

```
GET <remote subnet-ref>
```

Response

Representation of specified remote subnet

PUBLIC IP RESOURCE

Public IP resources represent IP addresses that can be used to route network traffic to VMs over the internet, or as an endpoint for VPNs. In the API, you make calls to Public IPs by referencing "ips."

PUBLIC IP MODEL

This resource represents the connection that can be created between two networks in different configurations. The element name is "tunnel" and the following fields are defined:

Field Name	Type	Access	Description
id	Integer	ro	ID of the Public IP (same as address)
address	Integer	ro	Unique IP adresss
region	String	ro	Region where public IP is housed (US-West or US-East)
nics	Array	ro	Representation of attached network adapter, if any
vpn_id	String	ro	ID of attached VPN, if any

OPERATIONS ON PUBLIC IPS

The public IP is not a top-level resource, meaning a description of a single public IP cannot be called directly. Public IPs attached to a network interface (interfaces) are a nested resource of the interface.

GET LIST OF AVAILABLE PUBLIC IPS

Request

```
GET https://cloud.skytap.com/ips
```

Response

Description of available public IPs

ATTACH PUBLIC IP TO NETWORK ADAPTER

Request

```
POST https://cloud.skytap.com/vms/<vm-id>/interfaces/<interface-
id>/ips?ip=<public_ip_address>
```

Response

Public IP attached to specified network

VIEW PUBLIC IPS ATTACHED TO A NETWORK ADAPTER

Request

```
GET https://cloud.skytap.com/vms/<vm-ref>/interfaces/<interface-id>/ips
```

Response

List of attached Public IPs

DETACH PUBLIC IP FROM A NETWORK ADAPTER

Request

```
POST https://cloud.skytap.com/vms/<vm-id>/interfaces/<interface-
id>/ips/<public-ip-address/detach
```

Response

Newly detached Public IP

INTER-CONFIGURATION NETWORK ROUTE (‘TUNNEL’) RESOURCE

Tunnels are top-level resources that define a network route between two networks.

INTER-CONFIGURATION NETWORK ROUTE REFERENCE URI

The tunnel reference URI is as follows:

```
<tunnel-ref> :==  https://.../tunnels/<tunnel-id>
```

INTER-CONFIGURATION NETWORK ROUTE MODEL

This resource represents the connection that can be created between two networks in different configurations. The element name is “tunnel” and the following fields are defined:

Field Name	Type	Access	Description
id	String	ro	identifier
status	String	ro	Status of the tunnel
source_network	network	ro	Source network from where the connection was initiated
target_network	network	ro	Target network to which the connection was made

INTER-CONFIGURATION NETWORK ROUTE SEMANTICS

This resource defines the network route that is set up when the user connects a network in one configuration to a network in another configuration.

SOURCE NETWORK, TARGET NETWORK

Source network refers to the ‘network’ resource that ‘initiated’ the connection to the target network. The ‘tunnelable’ attribute of the target network should be set to True for this tunnel to be created. The ‘tunnelable’ property of source network is inconsequential for this tunnel.

OPERATIONS ON INTER- CONFIGURATION NETWORK ROUTES

GET A TUNNEL DESCRIPTION

Request

```
GET <tunnel-ref>
```

Response

Identified tunnel resource

CREATE A TUNNEL

Request

```
POST /tunnels?source_network_id=<id>&target_network_id=<id>
```

Response

New tunnel resource

DELETE A TUNNEL

Request

```
DELETE <tunnel-ref>
```

Response

Status

VIRTUAL NETWORK INTERFACE RESOURCE

A **virtual network interface** consists of a virtualized network adapter in a virtual machine. It is a component of a virtual machine, so operations on network interfaces are implicitly operations on the containing configuration.

At the same time, the interface is attached to a virtual network.

NETWORK INTERFACE REFERENCE URI

Network interfaces are properties of virtual machines, so references to network interfaces are extensions of references to virtual machines. The network interface reference URI is as follows:

```
<network-interface-ref> :== <vm-ref>/interfaces/<interface-id>
```

NETWORK INTERFACE RESOURCE MODEL

The element name is "interface" and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	identifier
ip	IP address	rw	ip address (e.g. 10.1.0.37)
hostname	string	rw	host name

mac	string	rw	mac address (e.g. 00:1e:c2:c3:52:61)
services	service[]	rw	array of Published Services

NETWORK INTERFACE SEMANTICS

Generally, a Network Interface is a virtualization of a real NIC, and is logically contained in a VM and attached to a network.

IP, HOSTNAME, MAC

The ip, hostname and mac address attributes are all modifiable.

Internally, Skytap Cloud uses DHCP to provision ip and hostname to the interface based on the mac address, and Skytap Cloud will not assign the same mac address or ip address to multiple interfaces.

SERVICES

See the section on Published Services.

NETWORK_REF

This is a reference to the virtual network to which this interface is attached.

OPERATIONS ON NETWORK INTERFACE RESOURCES

GET NETWORK INTERFACE DESCRIPTION

Request

```
GET <network-interface-ref>
```

Response

Identified network interface element

UPDATE NETWORK INTERFACE

Request

```
PUT <network-interface-ref>?<attr-name>=<attr-value>
```

Response

Updated network interface element

PUBLISHED SERVICE RESOURCE

Generally, a **published service** represents a binding of a port on a virtual network interface to an IP and port that is routable and accessible from the public internet. This mechanism is used to selectively expose ports on the guest to the public internet.

Published services exist and are managed as aspects of network interfaces—that is, as part of the overall configuration element.

PUBLISHED SERVICE REFERENCE URI

Published services are elements within Network Interfaces, so the reference to a published service is an extension of the relevant Network Interface. The published service reference URI is as follows:

```
        <published-service-ref> :== <network-interface-ref>/services/<service-id>
```

PUBLISHED SERVICE CONTENT MODEL

The element name is "service" and the following fields are defined:

Field Name	Type	Access	Description
id	string	Ro	identifier
internal_port	integer	Rw	port exposed on interface
external_ip	ip address	Ro	exposed routable ip
external_port	integer	Ro	exposed port on external_ip

PUBLISHED SERVICE SEMANTICS

INTERNAL_PORT

The internal_port defines the port on the interface that is bound. Typically this will be dictated by standard usage—e.g. port 80 for http traffic, port 22 for ssh, etc.

EXTERNAL_IP, EXTERNAL_PORT

Connections to this IP/port are routed to the internal port in the guest VM. The external IP and port are selected by Skytap Cloud.

OPERATIONS ON PUBLISHED SERVICE RESOURCES

LIST PUBLISHED SERVICES

Request

```
        GET <network-interface-ref>/services
```

Response

A list of published-services, in the form:

```
<services>

        <service>…</service>

        …
```

```
</services>
```

DESCRIBE PUBLISHED SERVICE

Request

```
GET <published-service-ref>
```

Response

A published-service element.

ADD A PUBLISHED SERVICE FOR A GIVEN PORT

Request

Where “X” is equal to the port number:

```
POST <interface-ref>/services?port=X
```

Response

XML representation of newly created port.

DELETING A PUBLISHED SERVICE

Request

Where “X” is equal to the port number:

```
DELETE <interface-ref>/services/X
```

Reponse

Status

ASSET RESOURCE

Files uploaded to your account are called assets, and can be downloaded onto VMs in your account, as well as shared with other users via Projects. Assets are top-level elements in the API data model.

In the current release of the API, Assets can only be viewed, referenced and ownership managed, but not created, modified, or deleted.

ASSET REFERENCE URI

The customer asset reference URI is as follows:

```
<asset-ref> :== https://.../assets/<asset-id>
```

ASSET RESOURCE MODEL

The element name is “asset” and the following fields are defined:

Field Name	Type	Access	Description
Id	string	ro	identifier
Name	string	ro	human friendly name
Size	integer	ro	size (in GB?)
Owner	URI	rw	user-ref for owner
Region	string	ro	Region where the asset is housed

ASSET SEMANTICS

OWNER

Like other resources that may be owned, a customer asset belongs to exactly one user, and updating the owner field with a new user-reference will transfer ownership (subject to permission and quota constraints).

REGION

The asset's region is defined at asset creation, and lists where the asset is housed. This will be either "US-West" or "US-East".

OPERATIONS ON ASSET RESOURCES

LIST ASSETS

Request

```
GET /assets[?attr=val]
```

Response

A list of assets, optionally filtered on attribute. Data is not returned.

GET ASSET DESCRIPTION

Request

```
GET <asset-ref>
```

Response

An asset-element

CHANGE OWNER OF ASSET

This is an instance of changing an attribute. The operation will fail if the new owner not have sufficient quota to “accept” the asset.

Request

```
PUT <asset-ref>?owner=<user-ref>
```

Response

Updated asset element

USER RESOURCE

A **user** resource is used to capture authentication information (login name and password), to serve as a context for settings that span configurations and usage sessions, and to act as “owner” of resources in the system.

Users are top-level elements in the API data model.

USER REFERENCE URI

The user reference URI is as follows:

```
<user-ref> :== https://.../users/<user-id>
```

USER RESOURCE MODEL

The element name is “user” and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	identifier
url	URI	ro	URI reference for resource
login_name	string	ro	login name
first_name	string	rw	first name for user
last_name	string	rw	last name for user
title	string	rw	optional text container field for user title (if any)
password	string	wo	write-only (always displays blank)
email	string	rw	email address for user
sra_compression	integer	rw	SRA compression
account_role	string	rw	account role for user
quotas	quota[]	rw	array of Quota resources
configurations	Configuration[]	ro	array of Configuration resources
templates	Template[]	ro	array of Template resources
assets	Asset[]	ro	array of Asset references
can import	boolean	rw	determines if user can import VMs
can export	boolean	rw	determines if user can export VMs
sso_enabled	boolean	rw	Determines if account is enabled for SSO. NOTE: will only appear if your account is enabled for SSO and sso_enabled is set to true.

When a list of users is returned, only the first three elements are returned for each user.

USER SEMANTICS

LOGIN_NAME, PASSWORD

These fields are the basis for authentication to Skytap Cloud.

The password field is write-only—a new password may be set by assigning to this field, but it will always be returned as the special value “**********” (10 asterisks).

If that special value is sent to the server as the value of the password element, it is ignored—the previously set password is unchanged.

FIRST_NAME, LAST_NAME

Strings used for display in the UI. Not validated for uniqueness, etc.

EMAIL

Email address to which notifications may be sent. Not validated at submit time.

SRA_COMPRESSION

An integer (1-7) that defines the amount of compression used to encode SRA data streams. Higher values of compression can improve responsiveness of the SRA client but decrease visual quality.

ACCOUNT_ROLE

A user’s account role defines their access to the full range of Skytap features and settings. The four possible users roles are “restricted user,” “standard user,” “user manager,” and “admin.” For more information on user roles, see https://cloud.skytap.com/docs/index.php/Skytap_Basics#Creating_user_accounts.

CAN_IMPORT, CAN_EXPORT

These attributes determine if a restricted user, standard user, or user manager can import or export VMs. Setting the value to 1 enables the user to see and access an "import" or "export" tab in the Skytap Cloud web interface, allowing the user to import or export VMs. By default, these values are set to 0, and the user will not be able to import or export VMs.

Note that users with an account role of administrator will have these values locked to 1.

SSO_ENABLED

This field will only appear if your customer account is enabled for single sign-on, or SSO. If set to false, this is a standard Skytap Cloud user account; if set to true, the account will use SSO to log in to Skytap Cloud.

OPERATIONS ON USER RESOURCES

CREATE USER

When creating a new user, only the “login_name,” “password,” and “email” elements are required—all others may be empty, and can be modified later.

Request

```
POST /users
```

```
<user> … </user>
```

The user object is contained in the HTTP body.

Response

The new user element.

LIST USERS

Request

```
GET /users
```

Response

A list of user elements (short form).

GET USER DESCRIPTION

Request

```
GET <user-ref>
```

Response

A user element.

UPDATE USER DESCRIPTION

Not all attributes are modifiable.

Request

```
PUT <user-ref>?<attr-name>=<attr-value>
```

Response

Updated user element.

DELETE USER

Request

```
DELETE <user-ref>[?transfer_user_id=<user-id>]
```

Response

Status.

When deleting a user, a transfer_user_id must be specified. Any resources owned by the user will be assigned to the user specified. The transfer user must have sufficient quota to absorb the resources from the deleted user; otherwise, the transfer use will receive an error message and no resources will be transferred or deleted.

ASSIGN RESOURCE OWNERSHIP TO USER

Users are owners of various system resources: configurations, templates, and assets. The user content model allows review of resources owned by the user. Assignment of ownership, however, is done by updating the "owner" attribute of the respective resource.

See details of this attribute, see the sections on templates, configurations, and assets.

USER QUOTA RESOURCE

User Quotas are effectively complex attributes of users, and are properly contained within them with respect to the content model.

USER QUOTA REFERENCE URI

User Quotas exist only in the context of a user object, so references to them are relative to the user. The user quota resource URI is as follows:

```
<quota-ref> :== <user-ref>/quotas/<quota-type>
```

For instance

```
https://cloud.skytap.com/users/999/quotas/concurrent_storage_size
```

USER QUOTA RESOURCE MODEL

The element name is "quota" and the following fields are defined:

Field Name	Type	Access	Description
quota_type	string	Ro	see types below
Units	string	Ro	type specific
limit	integer	Rw	usage limit for user
usage	integer	Ro	current usage against quota

Currently there are several resources for which quotas are defined:

Quota Type	Units	Description	When quota reached
concurrent_storage_size	One of (MB, GB)	Storage limits, units specified in units element	
concurrent_svms	integer	Count of Skytap Virtual Machines that may be used at the same time.	
cumulative_svms	integer	Count of hours of SVM usage (i.e., SVM hours)	

USER QUOTA SEMANTICS

QUOTA_TYPE, UNITS

The 'quota_type' field specifies what kind of resource the quota is measuring (see the list above) and 'units' reflects that resources' current usage.

Note that it is not required for every user to have quota defined for all or even any quota type: in case a user does *not* have a quota of a particular type, the quota for the customer account is used by default.

LIMITS

A settable limit for the particular quota type.

What happens when the quota is reached depends on the type of quota:

- When the SVM Hours quota is reached, all running virtual machines are suspended. Using more SVM Hours requires an increase in the quota limit.
- When the SVM quota is reached, no further virtual machines can be started. Stopping running virtual machines "frees up" SVMs and allows more to be started.
- When the storage quota is reached, no more templates or configurations may be created. Deleting existing templates or configurations allows more to be made.

USAGE

A report of the amount of quota resource in use or consumed (depending on the resource).

OPERATIONS ON USER QUOTAS

CREATE USER QUOTA

When creating a quota object, the "type" and "limit" fields are required, while the "units" and "usage" fields are ignored.

Request

```
POST <user-ref>/quotas
```

The quota object is contained in the HTTP body.

Response

The new user element.

LIST USER QUOTAS

Request

```
GET <user-ref>/quotas
```

Response

List of quotas defined for user.

UPDATE USER QUOTA

Request

```
PUT <user-ref>/quotas/<resource>?limit=<value>
```

Response

Status

DELETE USER QUOTA

Request

```
DELETE <user-ref>/quotas/<resource>
```

Response

Status

PUBLISHED SET RESOURCE

A **published set** is a complex resource nested within a Configuration object, and represents one or more published URIs. A configuration can have no published sets, one published set, or multiple published sets.

If a configuration has one or more published sets, individual VMs belonging to the configuration are assigned URIs that provide direct access to the VM's desktops via a standard browser, without requiring creation of a user account in Skytap Cloud. Published sets provide an easy way to provision user access to the VMs.

In the API, these URIs appear as elements of the "configuration" content model, named "<publish_set>."

In the Skytap Cloud User Interface, these URIs are accessed through the Published URIs menu on the Configurations page.

PUBLISHED SET REFERENCE URI

The published set reference URI is as follows:

```
<publish-set-ref> :== <configuration-ref>/publish_sets/<publish-set-id>
```

PUBLISHED SET RESOURCE MODEL

The element name is "publish set" and the following fields are defined:

Field Name	Type	Access	Description
Id	string	ro	unique ID of published set
name	string	rw	descriptive name of published set: does not need to be unique
url	string	ro	URL of resource for UI and API access
publish_set_type	one of (single_url, multiple_url)	rw	type of URL published for associated VM or collection of VMs
start_time	time-of-day	rw	start of daily access window
end_time	time-of-day	rw	end of daily access window
time_zone	time zone	rw	time zone for access window
password	string	wo	Password
desktops_url	string	ro	anonymous access URL for single_url published sets
vms	vm[]	1	collection of VM descriptors including VM references associated with published set

PUBLISHED SET SEMANTICS

NAME

This is a descriptive name for the published set, provided so that the account user can easily identify the published set. For single_url published sets where the SRA viewer displays one or more VMs in a multi-VM frame, this string is also present in the browser's title bar for ease of identification.

PUBLISH_SET_TYPE

Two specific types of Published set can be configured:

Published set type	Description
single_url	Each VM assigned to the published set is accessible through a single shared URL: navigating to this URL in a browser will display the SRA viewer console (much like the account user SRA console) listing all associated VMs in a single window.
multiple_url	Each VM associated with the published set is accessible through its own unique URL: navigating to this URL in a browser will display the standalone SRA viewer applet in the window, allowing access to just that single VM.

START_TIME, END_TIME, TIME_ZONE

Access to the VM via the URL may be constrained to specific hours of the day: an access window is defined by the <start_time> and <end_time> elements.

A <time_zone> may be specified to qualify the access window. If this is not provided, the API caller's default time zone is used.

The times are expressed in HH:MM format, for hours 0:23 and minutes 0:59.

Time zones are text strings selected from the list included in Appendix 3: Time Zones.

When defining an access window, both start_time and end_time elements must be supplied. When modifying an existing access window, only the element being changed is required.

To delete an access window, specify empty strings for the start_time and end_time.

PASSWORD

A password may be included in a Published URL, in which case the password must be supplied by the user upon accessing the URL.

It's not possible to retrieve a password using this mechanism, but it is possible to tell whether one has been set or not. If no password is set for the access, the returned element will be empty. If a password has been set, the value returned here will be a string of 10 asterisks:

```
**********
```

If this special value is *sent* as the value of the password element, it is ignored; a previously set password will not be changed.

Setting a password to an empty string effectively disables the password and allows access to the published set without a password.

DESKTOPS_URL, DESKTOP_URL

This is the single URL used to anonymously access all VMs associated with a single_url published set. This value will not be present for a multiple URL published set, which will instead define multiple individual desktop_url values in the VM's array.

VMS

This is an array consisting of 'wait VM' elements, which are complex descriptors that include VM references and define the following per-VM properties:

Field Name	Type	Access	Description
vm_ref	string	rw	REST URI of associated VM resource
access	one of (view_only, use, run_and_use)	rw	Specifies the VM's access mode when accessed via the containing published set
desktop_url	string	ro	access URL for multiple_url published sets

The semantics of the desktop_url property are equivalent to the published set's desktops_url, except that they are applicable only to an individual VM for a multiple_url published set. The access mode value's semantics are as follows:

Access	Description
do_not_publish	The URL will not be available to the user in any way.
view_only	User can view the VM through the URL. No mouse or keyboard control is allowed, and desktop resizing is disabled.
Use	User can view the VM through the URL and interact with it using a mouse or keyboard. Desktop resizing is allowed.
run_and_use	User can view the VM through the URL and interact with it using a mouse or keyboard Desktop resizing is allowed, and the user can also run the machine if it is in a stopped or suspended state.

OPERATIONS ON PUBLISHED SET RESOURCES

CREATE PUBLISHED SET

The body of the HTTP request contains the XML representation of the intended published set object.

When creating a published set resource, you're required to provide a "name," "publish_set_type," and "password." Optionally, you can pair a "start_time"/"end_time" with "time_zone."

Request

```
POST <configuration-ref>/publish_sets

<publish_set> … </publish_set>
```

Response

The content model representation for the newly created published set object.

READ PUBLISHED SET

All fields will be returned, even if they have nil contents. Note that the password will be returned as an empty element if not configured or as the obfuscated value "**********" if it is set.

Request

```
GET <publish-set-ref>
```

Returns

The reference model in the body of the response.

UPDATE PUBLISHED SET

Request

```
PUT <publish-set-ref>

<publish-set> … </publish-set>
```

Returns

The updated representation of the published set.

DELETE PUBLISHED SET

Request

```
DELETE <publish-set-ref>
```

Returns

Status code

NOTIFICATION RULE RESOURCE

The **notification rule** resource refers to a rule you can set up to be alerted about certain events happening in your account. There are two kinds of notification rules that can be created in Skytap:

- **Account level rules:** These rules are set up to monito events at an account-wide scope. This scope includes events occurring in your account, e.g. SVM hours usage for the account. It also includes events occurring for a user in your account, e.g. SVM hours usage for a specific user or groups of users. Only Administrators can set up these account level rules.
- **User level rules:** These rules are set up to monitor events occurring for a user in their account. These rules can be created by a user only for their own account.

For a detailed description of the feature, please refer to the Notifications section in Skytap Documentation.

NOTIFICATION RULE REFERENCE URI

The template reference URI is as follows:

- **Account level rules:**

```
<notification-rule-ref> :==  https://.../notifications/<notification-id>
```

- **User level rules:**

```
<notification-rule-ref> :==  https://.../account/notifications/<notification-
id>
```

NOTIFICATION RULE RESOURCE MODEL

The element name is “notification_rules” and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	identifier
threshold_amount	string	rw	string defining the threshold level
rule_type	string	rw	notification event being monitored
scope	string	rw	scope for monitoring the event
subjects	subject[]	rw	array of users to be monitored, if the scope is ‘selected-users’
recipients	recipient[]	rw	array of recipients for the notification

NOTIFICATION RULE SEMANTICS

ID

Unique identifier for a notification rule.

THRESHOLD AMOUNT

Integer that represents the threshold values for the notification rule. For ‘VMUptimeNotificationRule’ this value represents the number of hours. For all other rules this value represents a percentage of usage.

RULE TYPE

Each rule monitors a type of event in the system. The following is the list of supported rule types and the events being monitored for each type.

Rule Type	Description
SVMHoursNotificationRule	SVM hours usage
StorageNotificationRule	Storage usage
ConcurrentVMNotificationRule	Number of concurrently running SVMs
IPAddressesUsedNotificationRule	Number of public IP addresses in use
NetworksUsedNotificationRule	Number of networks in use
VMUptimeNotificationRule	VM running for more than ‘N’ number of hours

SCOPE

A string that represents the scope at which events are being monitored. Scope is needed only for **Account Level Rules**. The following strings are supported for defining scope:

Scope	Description
customer-quota	To monitor the Subscription for the account
customer-limit	To monitor the Limits for the account
all-users	To monitor all users
selected-users	To monitor selected users, or groups of users
rule-owner	To monitor the user who created the rule. This string is supported only for **User Level Rules**

All scopes are not valid for every rule type. The following table lists the scopes that are valid for each rule type.

	customer-	customer-	all-users	selected-

	quota	limit		users
SVMHoursNotificationRule	✓	✓	✓	✓
StorageNotificationRule	✓	✓	✓	✓
ConcurrentVMNotificationRule		✓	✓	✓
IPAddressesUsedNotificationRule		✓		
NetworksUsedNotificationRule		✓		
VMUptimeNotificationRule			✓	✓

SUBJECTS

When scope is defined as **selected-users**, this array represents the list of users being monitored. The XML representation of one element (subject) of the array is given below:

```
<subject>
  <type>User</type>
  <id>11</id>
  <name>Joe User</name>
</subject>
```

Where,

- `type`: Represents the type of `subject`. Supported `subject` types are:
  - `User`: An individual user
  - `Group`: A group of users defined in the account
- `id`: is the Identifier for the `User` or `Group` resource.
- `name`: is the full name of the user or group resource.

RECIPIENTS

This is an array of users, who will be notified when the rule is triggered. Recipients are needed only for **Account Level Rules.**

The XML representation of one element (recipient) of the array is given below:

```
<recipient>
  <type>MetaRecipient</type>
  <id>1</id>
  <name>Resource Consumer</name>
</recipient>
```

Where,

- `type`: Represents the type of `recipient`. The supported `recipient` types are:

Recipient Type	Description
User	An individual user in the account
Group	A group of users defined in the account
MetaRecipient	Logical representation of set of users in the account. E.g. User who owns the resource

	being monitored or a group of all Administrators, etc.

- `id`: is the Identifier for the `User` or `Group` resource. For `MetaRecipient, id` represents the logical identifier of the logical group of users in the account

MetaRecipient Names	Id
Resource Consumer	1
All Admins	2
All User Managers	3
All Standard Users	4

- `name`: is the full name of the user or group resource. For `MetaRecipient` the names supported are:

MetaRecipient Names	Description
Resource Consumer	Consumer of the resource being monitored. E.g. owner of a VM
All Admins	Logical group of all Administrators in the account
All User Managers	Logical group of all User Managers in the account
All Standard Users	Logical group of all Standard Users in the account

OPERATIONS ON NOTIFICATION RULE RESOURCE

DESCRIBE NOTIFICATION RULE

Request

```
GET <notification-rule-ref>
```

Response

Notification element

DELETE NOTIFICATION RULE

Request

```
DELETE <notification-rule-ref>
```

Response

Status only

LIST NOTIFICATION RULES

Request

For Account Level Rules

```
GET /notifications
```

For User Level Rules

```
GET /account/notifications
```

Response

Returns a list of notification rules set for the account or for the user, depending on the resource being requested.

CREATE NEW NOTIFICATION RULE

Request

For Account Level Rules

```
POST /notifications
```

For User Level Rules

```
POST /account/notifications
```

Sample XML Body (for Account Level Rule)

```
<?xml version="1.0" encoding="UTF-8"?>
<notification_rule>
  <threshold_amount>100</threshold_amount>
  <rule_type>StorageNotificationRule</rule_type>
  <scope>customer-quota</scope>
  <recipients type="array">
    <recipient>
      <type>MetaRecipient</type>
      <id>1</id>
    </recipient>
    <recipient>
      <type>MetaRecipient</type>
      <id>2</id>
    </recipient>
  </recipients>
</notification_rule>
```

Sample XML Body (for User Level Rule)

```
<?xml version="1.0" encoding="UTF-8"?>
<notification_rule>
  <threshold_amount>100</threshold_amount>
  <rule_type>StorageNotificationRule</rule_type>
</notification_rule>
```

Response

Status and the new rule in the XML body

- When creating rules, only ids are required for `subjects` and `recipients,` names are not required.
- For creating **User Level Rules,** `scope` and `recipient` elements are not needed in the XML body.

IMPORT RESOURCE

An "import" represents a single import job. Each import job will import a single virtual machine image and place it into a new template containing the VM, a single network, and a single network adapter.

IMPORT URI

The import URI is as follows:

`<import-ref> = https://cloud.skytap.com/imports/<id>`

IMPORT RESOURCE MODEL

The element name is "import" and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	Reference ID for import job.
ftp_user_name	string	ro	User name for FTP login.
ftp_password	string	ro	Password for FTP login.
ftp_host	string	ro	FTP host name.
ftp_url	string	ro	FTP URL in specific format (see below).
template_name	string	rw	Name of template to be created.
template_description	string	rw	Template description, MUST be specified.

template_url	string	ro	URL where the template will be located.
status	string	rw	Status of import job (see description).
network_domain	string	rw	Domain given to template network.
network_subnet	string	rw	Subnet given to template network.
interface_ip	string	rw	IP of network interface.
interface_hostname	string	rw	Hostname of network interface.
credentials	string	rw	Free-text field to store access credentials.
errors	string	ro	Describes error when "status"=error.
expiration_date	string	ro	Date import job will be deleted.
user	string	rw	An array defining user parameters.
region	string	rw	Region where the template will be stored

IMPORT SEMANTICS

FTP_URL

The full FTP path to the file. It must be in the following format:

```
ftp://<ftp-username-ref>:<ftp-password-ref>@<hostname-ref>/upload/
```

TEMPLATE_URL

This field does not exist until the "status" field equals "complete." At this point the URL will take you to the newly created template.

STATUS

The "status" field can equal one of four states: "new," "processing," "error," and "complete." The status will be "new" upon initial creation; to start the import process, the status must be changed to "processing." If an error occurs, the status will read "error" and the error will be detailed in the "error" field. When the import is complete, the status will read "complete."

USER

The user array provides information on the user that created the import.

id	string	ro	User reference ID.
url	string	ro	URL of user's page in Skytap.
login	string	ro	User's login name
first_name	string	ro	User's first name.
last_name	string	ro	User's last name.

REGION

Skytap houses templates and configurations in one of two regions: US-West and US-East. When you import a VM, you must specify which region the resulting template will be created in. You should aim to house it in the location closest to the majority of your users. If no region is specified, the import will default to US-West.

For more information on regions, see https://cloud.skytap.com/docs/index.php/Understanding_Regions.

OPERATIONS ON IMPORT RESOURCES

CREATE A NEW IMPORT

To create an import job, you must make a POST to <import-ref> with the six required paremeters: **template_name**, **template_description**, **network_domain**, **network_subnet**, **interface_ip** and **interface_hostname**. These will define the settings of the newly created template.

Request

```
POST <import-ref> = <template name>&template_description=<template
description>&network_domain=<network domain>&network_subnet=<network
subnet>&interface_ip=<interface
ip>&interface_hostname=<hostname>&region=<region>
```

Response

XML representation of your new import.

CHANGE IMPORT STATUS

Request

```
PUT <status-ref>
```

Response

XML representation of resource with updated status field.

DESCRIBE IMPORT STATUS

Request

```
GET <status-ref>
```

Response

XML representation of status field.

EXPORT RESOURCE

An “export” represents a single import job. Each export starts with a VM contained in a template owned by the user.

EXPORT URI

The export URI is as follows:

```
<export-ref> = https://cloud.skytap.com/exports/<id>
```

EXPORT RESOURCE MODEL

The element name is "export" and the following fields are defined:

Field Name	Type	Access	Description
id	string	ro	Reference ID for the export job.
ftp_user_name	string	ro	User name for FTP login.
ftp_password	string	ro	Password for FTP login.
ftp_host	string	ro	Host name of FTP.
ftp_url	string	ro	Download location for VM image.
template_name	string	ro	Name of source template.
vm_name	string	ro	Name of source VM.
status	string	ro	Status of export job.
errors	string	ro	Describes error when status = error.
filename	string	ro	File name created by Skytap, present in FTP URL.
expiration_date	string	ro	Date when export job will be deleted.
template_url	string	ro	URL of source template.
vm_url	string	ro	URL of source VM.
user	string	ro	Array representing a user, same as in "import" resource.

EXPORT SEMANTICS

STATUS

The "status" field can equal one of three states: "processing," "error," and "complete." The status will be "new" upon initial creation; to start the import process, the status must be changed to "processing." If an error occurs, the status will read "error" and the error will be detailed in the "error" field. When the export is complete, the status will read "complete."

OPERATIONS ON EXPORT RESOURCES

CREATE AN EXPORT

The only required parameter to create an export is the ID of the VM you want to export.

Request

```
POST <export-ref>= <source vm_id>
```

Response

XML representation of newly created export.

CHANGE EXPORT STATUS

Request

```
PUT <status-ref>
```

Response

XML representation of resource with updated status field.

DESCRIBE EXPORT STATUS

Request

```
GET <status-ref>
```

Response

XML representation of status field.

SCHEDULES RESOURCE

The "schedules" resource represents a set of one or more automated actions applied to a configuration or template. This resource functions identically to the schedules created in the Schedules tab in Skytap's web interface. All operations that can be performed on schedules from the Schedules tab can also be performed and automated via operations on the schedules resource in the API. For more information about the actions you can automate with schedules, see Automating VMs with Schedules.

SCHEDULES REFERENCE URI

The schedules reference URI is as follows:

```
<schedule-ref> = cloud.skytap.com/schedule/<id>
```

SCHEDULES RESOURCE MODEL

The element name is “schedules” and the following fields are defined. Note that schedules can be assigned to both templates and configurations, and some fields are unique to each resource type. These will be covered in separate tables below. Keep in mind that when a schedule is applied to a template, a configuration will first be created from that template and all scheduled actions will apply to that configuration.

Field Name	Type	Access	Description
id	string	ro	unique identifier for schedule, i.e. "123"
url	string	ro	full url for schedule, i.e. "cloud.skytap.com/schedules/123"
title	string	rw	human-friendly title
time_zone	string	rw	string specifying schedule's time zone
start_at	string	rw	start date of schedule
end_at	string	rw	time when schedule stops or

			configuration is deleted (see below)
delete_at_end	boolean	rw	deletes configuration at end time if enabled
notify_user	boolean	rw	determines whether user receives e-mail notification of successful actions
executions	executions[]	ro	List of objects detailing a schedule's progress through scheduled actions
recurring_days	recurring_days[]	rw	list of days on which schedule recurs
actions	actions[]	rw	lists of scheduled actions that can recur
next_action_name	string	ro	next action that will occur when schedule is executed
next_action_time	string	ro	specifies time at which the next action will occur
user	object	ro	key-value pairs representing the schedule's owner

CONFIGURATION FIELDS

These fields only apply to schedules operating on configurations.

Field Name	**Type**	**Access**	**Description**
configuration_id	string	ro	configuration's ID
configuration_url	string	ro	configuration's URL
configuration_name	string	ro	user-friendly name of the configuration

TEMPLATE FIELDS

These fields only apply to schedules operation on templates.

Field Name	**Type**	**Access**	**Description**
template_id	string	ro	template's ID
template_url	string	ro	template's URL
template_name	string	ro	user-friendly name of template
new_configuration_name	string	rw	name of configuration created from template
append_timestamp_to_name	boolean	rw	if enabled, appends creation time to configuration name
vm_ids	vm_ids[]	rw	list of VM IDs for newly created configuration

SCHEDULES SEMANTICS

TITLE

A reference title for the schedule. Limited to 256 characters.

TIME_ZONE

A string specifying the time zone in which the scheduled action will run. These are entered as English strings, i.e. "Mumbai" or "Pacific Time (US & Canada)." For a list of supported time zones and their nominal UTC offsets, see Appendix 3: Time Zones.

START_AT

A string specifying when the schedule will execute. The format for the time is YYYY/MM/DD hh:mm, where *YYYY* is a four-digit year, *MM* is a two-digit month, *DD* is a two-digit day, *hh* is a two-digit hour an mm is a two-digit minute. An example would be: "2012/07/28 11:09"

Notice that there is no UTC offset, as the time zone is determined by the time_zone string.

END_AT

This string specifies the end date of a schedule (entered in the same format as the start time, above). This string is only relevant if the schedule is set to recur (see recurring_days) and/or is set to delete at the end of its operations (see delete_at_end, below). If set to recur, the end_at time will specify the date at which the schedule stops recurring. If delete_at_end is set to true, then the end_at date will determine the time that the schedule's configuration is deleted. If both recurrence and deletion are enabled, the schedule will both cease recurring and delete the configuration at this time.

DELETE_AT_END

If end_at is set, this Boolean determines whether the schedule's configuration is deleted when the time specified in end_at is reached.

NOTIFY_USER

This Boolean determines whether the schedule's owner receives an email after the completion of a successful action. The user will always be notified when an action fails to complete, even if notify_user is not enabled.

EXECUTIONS

"Executions" is an array of objects that collectively represent an action performed when the schedule runs. The fields identify the resource targeted by the action, the action to be performed, and the time when the action will start. Before a schedule has been run, the executions array will contain an empty list; once started, the array will detail the target of the next action and the action to be performed. For more details, see the[**Executions Resource]** section.

RECURRING_DAYS

"Recurring_days" is an array that lists the days on which the schedule will recur. If this array is blank, the schedule will run until its final operation has been performed, and then delete itself. By entering one or more days in this string (Sunday, Monday, etc.) the schedule will continue to exist and will recur on those dates until the time set in end_at.

ACTIONS

"Actions" is an array of objects comprising all schedules actions in a schedule, with the exception of actions that cannot recur (i.e. the creation or deletion of configurations). The allowable actions are run, suspend, shut down, power off, and save. Each object in the array has two attributes:

- **Type**: A string that is one of the legal action types: RUN, SUSPEND, SHUT DOWN, POWER OFF, and SAVE.
- **Offset**: An integer equal to the number of seconds after configuration start that this action will take place. For instance, if the action was scheduled to start 2 hours after the schedule was executed, the offset would be 7200.

NEXT_ACTION_NAME

This string represents the next action that will execute as part of the schedule. This will normally be the name of one of the legal actions (above), but can also be "start" (if the schedule has not yet executed) or "delete" (if all scheduled actions have completed, end_at has a value, and delete_at_end is set as true).

NEXT_ACTION_TIME

A string representing the time when the next action (specified in next_action_name, above) will occur. Enter the date and time using the same format explained in [**start-at**].

USER

This is a list of key-value pairs representing the owner of the schedule. It contains the following fields:

- id: the user's account name
- url: the URL of the user's details page
- login_name: the name used to log in to Skytap Cloud
- first_name: the user's first name
- last_name: the user's last name
- email: the email address registered with Skytap Cloud

CONFIGURATION FIELDS

CONFIGURATION_ID

The numerical ID of the configuration the schedule is applied to.

CONFIGURATION_URL

The URL of the configuration the schedule is applied to, ending with the ID number. If the ID is 123, the URL will be https://cloud.skytap.com/configurations/123.

CONFIGURATION_NAME

A string representing the name of the configuration the schedule is applied to.

TEMPLATE FIELDS

TEMPLATE_ID

The numerical ID of the template that the schedule is applied to.

TEMPLATE_URL

The URL of the template the schedule is applied to, ending with the ID number. If the ID is 123, the URL will be https://cloud.skytap.com/configurations/123.

TEMPLATE_NAME

A string representing the name of the template the schedule is applied to.

NEW_CONFIGURATION_NAME

A string representing the name of the configuration that will be created from the target template once the schedule is started.

APPEND_TIMESTAMP_TO_NAME

A Boolean that determines whether a timestamp will be appended to the new configuration's name. This timestamp will reflect the date and time that the configuration was created.

OPERATIONS ON SCHEDULES RESOURCE

CREATE A NEW SCHEDULE

To create a schedule, you'll need to make a POST to /schedules with the following fields defined:

- **Title**
- **Configuration ID** or **Template ID**
- A **Start At** time
- The **Time Zone** the schedule will reside in.

If you would like to assign an action to the schedule during the POST, you'll need to include an **Actions** array, with its two elements (**Type** and **Offset**) defined. If you provide this array in XML, you ***must*** specify a type attribute of "array" for **Actions**, e.g. `<actions type = "array">…</actions>`. Specifying this type is not necessary for JSON.

Note that you can only include a single action when POSTing a schedule; additional actions will have to be edited in after creation.

Additional, some XML programs the ampersand in time zones will need to be represented as an escape character (`&amp;`).

Below is an XML example of a new schedule. This schedule will create a new configuration, run it one hour after the start time, and suspend it 8 hours after the start time.

```
POST /schedules
<?xml version="1.0" encoding="UTF-8"?>

<schedule>
  <title>My New Schedule</title>
  <configuration-id>123456</configuration-id>
  <actions type="array">
    <action>
      <type>run</type>
      <offset>3600</offset>
    </action>
    <action>
      <type>suspend</type>
      <offset>28800</offset>
    </action>
  </actions>
 <start-at>2013/09/09 09:00</start-at>

 <time-zone>Pacific Time (US & Canada)</time-zone>

</schedule>
```

SCHEDULE COLLECTION RESOURCE

The "schedules" resource described above refers to a single schedule. However, you can also perform GET actions on a collection of schedules in order to view basic details about all of the schedules in your account.

To do this, you'll need to perform a GET command on /schedules without specifying the ID of a specific schedule. However, for the command to be valid, you'll also need to specify two additional attributes: **count** and **offset**, as in this command:

```
GET schedules?count10&offset=0
```

Count determines the number of schedules the GET command will return, while **offset** determines how far "down the list" the API will retrieve schedules from. For instance, in an account with 40 schedules, and GET command with a count of 10 and an offset of 0 would display the first 10 schedules created in the account, while a GET command with a count of 10 and an offset of 20 would display schedules 21-30.

If you perform a GET command without specifying count and offset, you'll receive a 302 error and be redirected to the sample command provided above.

NAVIGATING A SCHEDULES COLLECTION

After executing a valid GET command, the specified schedules will be displayed. In addition, the command will generate a **content range header.** If you are viewing only a portion of the total schedules in your account, it will also generate a **link header**.

CONTENT RANGE HEADER

The content ranger header lists the range of items you are viewing, along with the total number in your account. For instance, if your account had 40 schedules, and you chose to view schedules 10-20, the header will display as `items 10-20/40`. For more details on content range headers, see this HTTP specification.

LINK HEADER

If you perform a GET command with a **count** less than your total number of schedules, the collection will be paginated. A **link header** will provide links with different offsets; copying these will allow you to view schedules of a lower or higher number. For more details on link headers, see this HTTP specification.

EXECUTION RESOURCE

The "executions" resource represents an action performed on a configuration as part of a schedule. As such, executions is not a top-level API resource, but exists as an array under a "schedules" resource. The fields in this array identify the resource targeted by the action, the action to be performed, and the time when the action will start.

Field Name	Type	Access	Description
id	string	ro	the execution's id
url	string	ro	the execution's reference URL
schedule_id	string	ro	ID of the schedule execution belongs to
schedule_url	string	ro	URL of same schedule
configuration_id	string	ro	ID of configuration that this executions affects
configuration_url	string	ro	URL of same configuration
next_action	object	ro	structure detailing schedule's next action
user_id	string	ro	ID of execution's owner (if Admin)
user_url	string	ro	URL of execution's owner (if Admin)

EXECUTIONS SEMANTICS

NEXT_ACTION

Next_action is a structure that details the target and scheduled time of the next action that will take place as part of the execution. This has three fields:

- **Type**: The type of action that will be run. The following are legal actions: run, power off, save, suspend, shut down, create, delete.
- **Time**: A string representing the time the action will take place. The format for the time is YYYY/MM/DD hh:mm, where *YYYY* is a four-digit year, *MM* is a two-digit month, *DD* is a two-digit day, *hh* is a two-digit hour and *ss* is a two-digit second. An example would be: "2012/07/28 11:09."
- **Time_zone**: A string representing the time zone of the execution. This should be the same as the schedule's time zone.

USER_ID

This will display the ID of the execution's owner if that owner is an administrator; otherwise, it will be blank.

USER_URL

This will display the URL of the execution's owner if that owner is an administrator; otherwise, it will be blank.

USAGE REPORT RESOURCE

The "usage report" resource represents a report of all account resource usage within a designated period. When generating the report, You can select what usage metric you want to generate reports on, how you want the report to be displayed, and what time period you're interested in viewing usage from.

USAGE REPORT REFERENCE URI

The usage report reference URI is as follows:

```
        <usage-report-ref> = cloud.skytap.com/reports/<id>
```

USAGE REPORT RESOURCE MODEL

The element name is “usage_report” and the following fields are defined:

Field Name	Type	Access	Description
id	integer	ro	ID of usage report
start_date	string	rw	Start date for data collection period
end_date	string	rw	End date for data collection period
utc	boolean	rw	Determines if reports use local time zone or UTC.
resource_type	string	rw	Specifies reporting of storage or svm usage.
region	string	rw	Regions whose activity will be included in report.
aggregate_by	string	rw	time when schedule stops or configuration is deleted (see below)
group_by	string	rw	How usage data is grouped (see below)
results_format	string	rw	File format of the report. **Must be specified as "csv".**
ready	boolean	ro	Specifies whether the report is ready for access
url	string	ro	URL where the report can be accessed.

USAGE REPORT SEMANTICS

START_DATE, END_DATE

You must specify a start date and end date for the report using an accepted date format (as specified in Appendix 2: Formatting of Dates). All usage in this date range will be reported.

UTC

This Boolean determines if the specified start and end dates correspond to UTC, which is used for Skytap's billing cycle, or the local time zone of the user making the report. If not specified, this value will default to "false."

RESOURCE_TYPE

Either "storage" or "svms" can be specified as the resource type. If storage is set, storage usage will be reported. If SVMs is set, the report will include cumulative SMV usage as well as max concurrent SVM usage.

REGION

This field determines the region that usage data is collected from. By default, it is set to "all," but you can also specify a specific region (e.g. US-West, US-East).

GROUP_BY

Reports can be grouped by user, group, region, or raw, which determines how the report's data is organized:

- **user** sorts usage by users;
- **group** sorts usage by group, as well as the individual users within each group;
- **region** groups the usage data by region;
- **raw** will display the individual line items of raw usage data, allowing you to see every instance of both SVM or storage usage.

AGGREGATE_BY

Aggregation can be set to "month", "day", or "none". This determines the increments at which data is displayed. For example, chosing "month" for the resource type "storage" will show the amount of storage used each month for the designated time period. "None" displays activity as a lump sum in that period. Note that reports grouped by "raw" must choose "none" for aggregation, as usage will be displayed in each line item.

RESULTS_FORMAT

The file format the results will be displayed in. **This must be entered as "results_format=csv"** in order to generate the appropriate CSV file.

OPERATIONS ON USAGE REPORTS RESOURCE

GENERATE A USAGE REPORT

Request

```
POST cloud.skytap.com/reports
start_date=<start_date>&end_date=<end_date>&utc=<utc>&resource_type=<resource_t
ype>&region=<region>&group_by=<group_by>&aggregate_by=<aggregate_by>&results_fo
rmat=<results_format>
```

Response

A representation of the new report resource, with two additional fields included: "id" and "ready=false". While the report resource has been created, the results must be compiled before the report can be viewed. Once the report is done processing, the field "ready=false" will change to "ready=true" and a new field, "url", will appear. Making a GET to this URL will show the results of the report.

APPENDICES

APPENDIX 1: INTENDED USAGE AND RESTRICTIONS

Please see Skytap Terms of Use (http://www.skytap.com/terms-of-use) for details. By using the Skytap Cloud API, you signify your assent to the Terms of Use.

APPENDIX 2: FORMATTING OF DATES

The API accepts dates in multiple standard formats. Dates are exposed in the API in the format "YYYY/MM/dd HH:mm:ss". We also accept the ISO 8601 format, e.g. 2013-03-21T01:00:00.000Z. Note that the number of digits is fixed, so single-digit dates or months must be preceded by zeroes, as in "2013/03/09."

APPENDIX 3: TIME ZONES

In order to accommodate locales that observe daylight savings time or have other unique time adjustment policies, the system manages time zones symbolically rather than simply by offset from UTC.

The table below lists the supported time zones and their nominal UTC offsets:

Time Zone Name	OFFSET	Time Zone Name	OFFSET	Time Zone Name	OFFSET	Time Zone Name	OFFSET
International Date Line West	-11:00	Edinburgh	+00:00	Sofia	+02:00	Beijing	+08:00
Midway Island	-11:00	Lisbon	+00:00	Tallinn	+02:00	Chongqing	+08:00
Samoa	-11:00	London	+00:00	Vilnius	+02:00	Hong Kong	+08:00
Hawaii	-10:00	Monrovia	+00:00	Baghdad	+03:00	Irkutsk	+08:00
Alaska	-09:00	UTC	+00:00	Kuwait	+03:00	Kuala Lumpur	+08:00
Pacific Time (US & Canada)	-08:00	Amsterdam	+01:00	Moscow	+03:00	Perth	+08:00
Tijuana	-08:00	Belgrade	+01:00	Nairobi	+03:00	Singapore	+08:00
Arizona	-07:00	Berlin	+01:00	Riyadh	+03:00	Taipei	+08:00
Chihuahua	-07:00	Bern	+01:00	St. Petersburg	+03:00	Ulaan Bataar	+08:00
Mazatlan	-07:00	Bratislava	+01:00	Volgograd	+03:00	Urumqi	+08:00
Mountain Time (US & Canada)	-07:00	Brussels	+01:00	Tehran	+03:30	Osaka	+09:00
Central America	-06:00	Budapest	+01:00	Abu Dhabi	+04:00	Sapporo	+09:00
Central Time (US & Canada)	-06:00	Copenhagen	+01:00	Baku	+04:00	Seoul	+09:00
Guadalajara	-06:00	Ljubljana	+01:00	Muscat	+04:00	Tokyo	+09:00
Mexico City	-06:00	Madrid	+01:00	Tbilisi	+04:00	Yakutsk	+09:00
Monterrey	-06:00	Paris	+01:00	Yerevan	+04:00	Adelaide	+09:30
Saskatchewan	-06:00	Prague	+01:00	Kabul	+04:30	Darwin	+09:30
Bogota	-05:00	Rome	+01:00	Ekaterinburg	+05:00	Brisbane	+10:00
Eastern Time (US & Canada)	-05:00	Sarajevo	+01:00	Islamabad	+05:00	Canberra	+10:00
Indiana (East)	-05:00	Skopje	+01:00	Karachi	+05:00	Guam	+10:00
Lima	-05:00	Stockholm	+01:00	Tashkent	+05:00	Hobart	+10:00

Quito	-05:00	Vienna	+01:00	Chennai	+05:30	Melbourne	+10:00
Atlantic Time (Canada)	-04:00	Warsaw	+01:00	Kolkata	+05:30	Port Moresby	+10:00
Caracas	-04:00	West Central Africa	+01:00	Mumbai	+05:30	Sydney	+10:00
La Paz	-04:00	Zagreb	+01:00	New Delhi	+05:30	Vladivostok	+10:00
Santiago	-04:00	Athens	+02:00	Kathmandu	+05:45	Magadan	+11:00
Newfoundland	-03:30	Bucharest	+02:00	Almaty	+06:00	New Caledonia	+11:00
Brasilia	-03:00	Cairo	+02:00	Astana	+06:00	Solomon Is.	+11:00
Buenos Aires	-03:00	Harare	+02:00	Dhaka	+06:00	Auckland	+12:00
Georgetown	-03:00	Helsinki	+02:00	Novosibirsk	+06:00	Fiji	+12:00
Greenland	-03:00	Istanbul	+02:00	Sri Jayawardenepura	+06:00	Kamchatka	+12:00
Mid-Atlantic	-02:00	Jerusalem	+02:00	Rangoon	+06:30	Marshall Is.	+12:00
Azores	-01:00	Kyev	+02:00	Bangkok	+07:00	Wellington	+12:00
Cape Verde Is.	-01:00	Minsk	+02:00	Hanoi	+07:00	Nuku'alofa	+13:00
Casablanca	+00:00	Pretoria	+02:00	Jakarta	+07:00		
Dublin	+00:00	Riga	+02:00	Krasnoyarsk	+07:00		

APPENDIX 4: CHANGE LOG

CHANGES IN REVISION 1.14

The 1.14 release includes the following documentation changes:

- Added Appendix 2: Formatting of Dates, with information on how dates should be entered into the API.
- Added "Usage Reports Resource" section.

The 1.14 release includes the following functional changes:

- The API now supports access to usage reports, as detailed in the "Usage Reports Resource" section.t

CHANGES IN REVISION 1.13

The 1.13 release includes the following documentation changes:

- Public IP Resource added to documentation

The 1.13 release includes the following functional changes:

- Users can be enabled for SSO at account creation; this is documented in the User Resource.

CHANGES IN REVISION 1.12

The 1.12 release includes the following documentation changes:

- Added "regions" field to various elements

CHANGES IN REVISION 1.10

The 1.10 release includes the following documentation changes:

- Added **Schedule Collection** section for details on viewing multiple schedules
- Revised introduction for greater clarity
- Documented support of JSON content type and covered relevant accept header and content type request.
- Clarified attributes for the actions array in **Schedules**.
- Added sample XML schedule to **Schedules** section.
- Update to **Templates** resource with command for saving published sets as part of template.

The 1.10 release includes the following functional changes:

- "can_import" and "can_export" added to user resource. These attributes determine if the user is able to import or export VMs.

CHANGES IN REVISION 1.9

The 1.9 release includes the following functional changes:

- Addition of **Schedules** resource and operations for the resource
- Addition of **Executions** resource and operations for the resource

CHANGES IN REVISION 1.8

The 1.8 release includes the following functional changes:

- Addition of **Import** resource and operations for the resource
- Addition of **Export** resource and operations for the resource

CHANGES IN REVISION 1.7

The 1.7 release includes the following functional changes:

- Addition of **Notification Rule** resource and operations for the resource
- Remove 'vnc_url' field from Virtual Machine Resource Model. This field is not supported in the API.

CHANGES IN REVISION 1.6

The 1.6 release includes the following documentation changes:

- Update to <configuration-ref> and <template-ref> XML tags used in the body of some PUT and POST operations. The correct tags are <configuration_id/> and <template_id/>. For these operations, id of templates and configuration should be used instead of the references.
- Inter-Configuration Network Route ('Tunnel') resource and operations for the resource
- Removed the requirement to have 'lockversion' on PUT operations on resources. 'lockversion' was originally envisioned to be a concurrency control mechanism. However, it resulted in an extra call users had to make for all PUT operations. We will consider other concurrency control mechanism for the API as we get more feedback on automation use cases.ƒ

CHANGES IN REVISION 1.5

The 1.5 release includes the following documentation changes:

- We support 1 to 8 CPUs per VM.
- Maximum RAM per VM is now 32,768 MB.
- Described the use of security tokens with Basic Auth

CHANGES IN REVISION 1.4

The 1.4 release documents incompatible changes to the Quota resource object.

CHANGES IN REVISION 1.3 R4

The 1.3 R4 release includes the following documentation changes:

- Deleting user requires **transfer_user_id** instead of **transfer_user** and it is no longer optional

CHANGES IN REVISION 1.3 R3

The 1.3 R3 release includes the following documentation changes:

- Introduced Skytap Cloud as product name.
- Reformatted the documentation.

CHANGES IN REVISION 1.3 R2

The 1.3 R2 release includes the following documentation changes:

- We support 1, 2, or 4 CPUs per VM.
- Maximum RAM per VM is now 8192 MB.

CHANGES IN REVISION 1.3

The 1.3 release includes the following change:

- Replacement of **Anonymous Access** resource with **published set** resource.

CHANGES IN REVISION 1.2 R2

The 1.2 R2 release includes the following documentation changes:

- Creating a user requires **email** in addition to the **login_name** and **password** elements specified in R1.
- R1 incorrectly specified that when creating a template the base configuration is specified through a **config-ref** instead of the correct **configuration-ref** element.
- Corrected the units for **storage** quota.

CHANGES IN REVISION 1.2

The 1.2 release includes the following functional changes:

- Addition of the **anonymous_access** element.

The 1.2 release included the following documentation changes:

- Improved documentation of resource model fields, syntax, and semantics.
- Documentation of the **anonymous_access** element.
- Documentation of error code 423.

CHANGES IN REVISION 1.1

The 1.1 release includes the following functional changes:

- Addition of the **user** and **user quota** elements.
- Addition of the **owner** attribute for **configuration** and **template** objects.
- Addition of the **suspend_on_idle** attribute for configuration objects.
- Addition of the **owner** and **public** elements for asset objects.

The 1.1 release included the following documentation changes:

- Expanded discussion of versioning and compatibility.
- Expanded discussion of error status codes.
- Documentation of the **halted** pseudo-state element for **configuration** and **virtual machine** objects.
- Inclusion of the **description** element for **template** objects.