OpenNebula 5.9 Integration Guide

Release 5.9.90

OpenNebula Systems

Nov 11, 2019

CONTENTS

1 System Interfaces **1**
- 1.1 Overview . . . 1
- 1.2 XML-RPC API . . . 2
- 1.3 Ruby OpenNebula Cloud API . . . 109
- 1.4 PyONE: Open Nebula Python Bindings . . . 112
- 1.5 Java OpenNebula Cloud API . . . 114
- 1.6 OneFlow Specification . . . 118
- 1.7 Go OpenNebula Cloud API . . . 144

2 Infrastructure Integration **147**
- 2.1 Overview . . . 147
- 2.2 Using Hooks . . . 147
- 2.3 Hook Manager Events . . . 156
- 2.4 Virtualization Driver . . . 158
- 2.5 Provision Driver . . . 166
- 2.6 Storage Driver . . . 168
- 2.7 Monitoring Driver . . . 180
- 2.8 Networking Driver . . . 184
- 2.9 Authentication Driver . . . 187
- 2.10 Cloud Bursting Driver . . . 189
- 2.11 Market Driver . . . 193
- 2.12 IPAM driver . . . 196
- 2.13 vCenter Driver . . . 200
- 2.14 NSX Driver . . . 208

3 References **211**
- 3.1 Overview . . . 211
- 3.2 Building from Source Code . . . 211
- 3.3 Build Dependencies . . . 217
- 3.4 Sunstone Development . . . 221

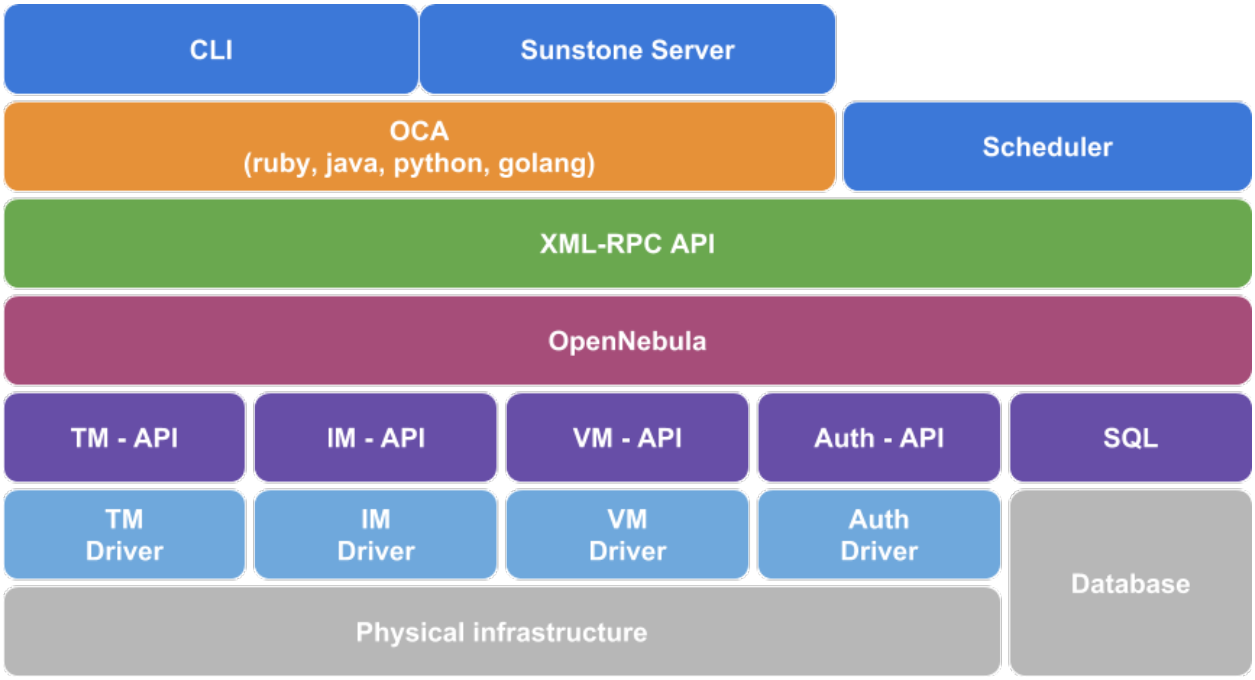
CHAPTER

ONE

SYSTEM INTERFACES

1.1 Overview

OpenNebula has been designed to be easily adapted to any infrastructure and easily extended with new components. The result is a modular system that can implement a variety of Cloud architectures and can interface with multiple datacenter services. In this Guide we review the main interfaces of OpenNebula, their use and give pointers to additional documentation for each one.



1.1.1 How Should I Read This Chapter

You should be reading this Chapter if you are trying to automate tasks in your deployed OpenNebula cloud, and you have already read all of the previous guides.

This Chapter introduces the OpenNebula system interfaces:

- The XML-RPC interface is the primary interface for OpenNebula, exposing all the functionality to interface the OpenNebula daemon. Through the XML-RPC interface you can control and manage any OpenNebula resource, including VMs, Virtual Networks, Images, Users, Hosts and Clusters. Use the XML-RPC interface if you are developing specialized libraries for Cloud applications or you need a low-level interface with the OpenNebula core. A full reference in the *XML-RPC reference Section*.

- The OpenNebula Cloud API provides a simplified and convenient way to interface with the OpenNebula core XMLRPC API. The OCA interfaces exposes the same functionality as that of the XML-RPC interface. OpenNebula includes four language bindings for OCA: *Ruby*, *JAVA*, *Golang* and *Python*.
- The *OpenNebula OneFlow API* is a RESTful service to create, control and monitor services composed of interconnected Virtual Machines with deployment dependencies between them.

After this Chapter, if you are interested in extending the OpenNebula functionality, you need to read the *Infrastructure Integration Chapter* to understand how OpenNebula relates with the different datacenter components.

1.1.2 Hypervisor Compatibility

All the Sections of this Chapter applies to both KVM and vCenter hypervisors.

1.2 XML-RPC API

This reference documentation describes the xml-rpc methods exposed by OpenNebula. Each description consists of the method name and the input and output values.

All xml-rpc responses share a common structure.

Type	Data Type	Description
OUT	Boolean	True or false whenever is successful or not.
OUT	String	If an error occurs this is the error message.
OUT	Int	Error code.

The output will always consist of three values. The first and third ones are fixed, but the second one will contain the String error message only in case of failure. If the method is successful, the returned value may be of another Data Type.

The Error Code will contain one of the following values:

Value	Code	Meaning
0x0000	SUCCESS	Success response.
0x0100	AUTHENTICATION	User could not be authenticated.
0x0200	AUTHORIZATION	User is not authorized to perform the requested action.
0x0400	NO_EXISTS	The requested resource does not exist.
0x0800	ACTION	Wrong state to perform action.
0x1000	XML_RPC_API	Wrong parameters, e.g. param should be -1 or -2, but -3 was received.
0x2000	INTERNAL	Internal error, e.g. the resource could not be loaded from the DB.
0x4000	ALLOCATE	The resource cannot be allocated.
0x8000	LOCKED	The resource is locked.

Note: All methods expect a session string associated to the connected user as the first parameter. It has to be formed with the contents of the ONE_AUTH file, which will be `<username>:<password>` with the default 'core' auth driver.

Note: Each XML-RPC request has to be authenticated and authorized. See the Auth Subsystem documentation for more information.

The information strings returned by the `one.*.info` methods are XML-formatted. The complete XML Schemas (XSD) reference is included at the end of this page. We encourage you to use the `-x` option of the command line interface to collect sample outputs from your own infrastructure.

The methods that accept XML templates require the root element to be TEMPLATE. For instance, this template:

```
NAME = abc
MEMORY = 1024
ATT1 = value1
```

Can be also given to OpenNebula with the following XML:

```
<TEMPLATE>
  <NAME>abc</NAME>
  <MEMORY>1024</MEMORY>
  <ATT1>value1</ATT1>
</TEMPLATE>
```

1.2.1 Authorization Requests Reference

OpenNebula features a CLI that wraps the XML-RPC requests. For each XML-RPC request, the session token is authenticated, and after that the Request Manager generates an authorization request that can include more than one operation. The following tables document these requests from the different CLI commands.

onevm

onevm command	XML-RPC Method	Auth. Request
deploy	one.vm.deploy	VM:ADMIN HOST:MANAGE
boot terminate suspend hold stop resume release poweroff reboot	one.vm.action	VM:MANAGE
resched unresched	one.vm.action	VM:ADMIN
migrate	one.vm.migrate	VM:ADMIN HOST:MANAGE
disk-saveas	one.vm.disksaveas	VM:MANAGE IMAGE:CREATE
disk-snapshot-create	one.vm.disksnapshotcreate	VM:MANAGE IMAGE:MANAGE
disk-snapshot-delete	one.vm.disksnapshotdelete	VM:MANAGE IMAGE:MANAGE
disk-snapshot-revert	one.vm.disksnapshotrevert	VM:MANAGE
disk-snapshot-rename	one.vm.disksnapshotrename	VM:MANAGE

Continued on next page

Table 1 – continued from previous page

onevm command	XML-RPC Method	Auth. Request
disk-attach	one.vm.attach	VM:MANAGE IMAGE:USE
disk-detach	one.vm.detach	VM:MANAGE
disk-resize	one.vm.diskresize	VM:MANAGE
nic-attach	one.vm.attachnic	VM:MANAGE NET:USE
nic-detach	one.vm.detachnic	VM:MANAGE
create	one.vm.allocate	VM:CREATE IMAGE:USE NET:USE
show	one.vm.info	VM:USE
chown chgrp	one.vm.chown	VM:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.vm.chmod	VM:<MANAGE/ADMIN>
rename	one.vm.rename	VM:MANAGE
snapshot-create	one.vm.snapshotcreate	VM:MANAGE
snapshot-delete	one.vm.snapshotdelete	VM:MANAGE
snapshot-revert	one.vm.snapshotrevert	VM:MANAGE
resize	one.vm.resize	VM:MANAGE
update	one.vm.update	VM:MANAGE
recover	one.vm.recover	VM:ADMIN
save	– (ruby method)	VM:MANAGE IMAGE:CREATE TEMPLATE:CREATE
updateconf	one.vm.updateconf	VM:MANAGE
list top	one.vmpool.info	VM:USE
list	one.vmpool.infoextended	VM:USE
–	one.vm.monitoring	VM:USE
lock	one.vm.lock	VM:MANAGE
unlock	one.vm.unlock	VM:MANAGE

Note: The **deploy** action requires the user issuing the command to have VM:ADMIN rights. This user will usually be the scheduler with the oneadmin credentials.

The scheduler deploys VMs to the Hosts over which the VM owner has MANAGE rights.

onetemplate

onetemplate command	XML-RPC Method	Auth. Request
update	one.template.update	TEMPLATE:MANAGE
instantiate	one.template.instantiate	TEMPLATE:USE [IMAGE:USE] [NET:USE]
create	one.template.allocate	TEMPLATE:CREATE
clone	one.template.clone	TEMPLATE:CREATE TEMPLATE:USE
delete	one.template.delete	TEMPLATE:MANAGE
show	one.template.info	TEMPLATE:USE
chown chgrp	one.template.chown	TEMPLATE:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.template.chmod	TEMPLATE:<MANAGE/ADMIN>
rename	one.template.rename	TEMPLATE:MANAGE
list top	one.templatepool.info	TEMPLATE:USE
lock	one.template.lock	TEMPLATE:MANAGE
unlock	one.template.unlock	TEMPLATE:MANAGE

onehost

onehost command	XML-RPC Method	Auth. Request
enable disable offline	one.host.status	HOST:ADMIN
update	one.host.update	HOST:ADMIN
create	one.host.allocate	HOST:CREATE [CLUSTER:ADMIN]
delete	one.host.delete	HOST:ADMIN
rename	one.host.rename	HOST:ADMIN
show	one.host.info	HOST:USE
list top	one.hostpool.info	HOST:USE

Warning: onehost sync is not performed by the core, it is done by the ruby command onehost.

onecluster

onecluster command	XML-RPC Method	Auth. Request
create	one.cluster.allocate	CLUSTER:CREATE
delete	one.cluster.delete	CLUSTER:ADMIN
update	one.cluster.update	CLUSTER:MANAGE
addhost	one.cluster.addhost	CLUSTER:ADMIN HOST:ADMIN
delhost	one.cluster.delhost	CLUSTER:ADMIN HOST:ADMIN
adddatastore	one.cluster.adddatastore	CLUSTER:ADMIN DATASTORE:ADMIN
deldatastore	one.cluster.deldatastore	CLUSTER:ADMIN DATASTORE:ADMIN
addvnet	one.cluster.addvnet	CLUSTER:ADMIN NET:ADMIN
delvnet	one.cluster.delvnet	CLUSTER:ADMIN NET:ADMIN
rename	one.cluster.rename	CLUSTER:MANAGE
show	one.cluster.info	CLUSTER:USE
list	one.clusterpool.info	CLUSTER:USE

onegroup

onegroup command	XML-RPC Method	Auth. Request
create	one.group.allocate	GROUP:CREATE
delete	one.group.delete	GROUP:ADMIN
show	one.group.info	GROUP:USE
update	one.group.update	GROUP:MANAGE
addadmin	one.group.addadmin	GROUP:MANAGE USER:MANAGE
deladmin	one.group.deladmin	GROUP:MANAGE USER:MANAGE
quota	one.group.quota	GROUP:ADMIN
list	one.grouppool.info	GROUP:USE
–	one.groupquota.info	–
defaultquota	one.groupquota.update	Ony for users in the `oneadmin` group

onevdc

onevdc command	XML-RPC Method	Auth. Request
create	one.vdc.allocate	VDC:CREATE
rename	one.vdc.rename	VDC:MANAGE
delete	one.vdc.delete	VDC:ADMIN
update	one.vdc.update	VDC:MANAGE
show	one.vdc.info	VDC:USE
list	one.vdcpool.info	VDC:USE
addgroup	one.vdc.addgroup	VDC:ADMIN GROUP:ADMIN
delgroup	one.vdc.delgroup	VDC:ADMIN GROUP:ADMIN
addcluster	one.vdc.addcluster	VDC:ADMIN CLUSTER:ADMIN ZONE:ADMIN
delcluster	one.vdc.delcluster	VDC:ADMIN CLUSTER:ADMIN ZONE:ADMIN
addhost	one.vdc.addhost	VDC:ADMIN HOST:ADMIN ZONE:ADMIN
delhost	one.vdc.delhost	VDC:ADMIN HOST:ADMIN ZONE:ADMIN
adddatastore	one.vdc.adddatastore	VDC:ADMIN DATASTORE:ADMIN ZONE:ADMIN
deldatastore	one.vdc.deldatastore	VDC:ADMIN DATASTORE:ADMIN ZONE:ADMIN
addvnet	one.vdc.addvnet	VDC:ADMIN NET:ADMIN ZONE:ADMIN
delvnet	one.vdc.delvnet	VDC:ADMIN NET:ADMIN ZONE:ADMIN

onevnet

onevnet command	XML-RPC Method	Auth. Request
addar	one.vn.add_ar	NET:ADMIN
rmar	one.vn.rm_ar	NET:ADMIN
free	one.vn.free_ar	NET:MANAGE
reserve	one.vn.reserve	NET:USE
updatear	one.vn.update_ar	NET:MANAGE
hold	one.vn.hold	NET:MANAGE
release	one.vn.release	NET:MANAGE
update	one.vn.update	NET:MANAGE
create	one.vn.allocate	NET:CREATE [CLUSTER:ADMIN]
delete	one.vn.delete	NET:MANAGE
show	one.vn.info	NET:USE
chown chgrp	one.vn.chown	NET:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.vn.chmod	NET:<MANAGE/ADMIN>
rename	one.vn.rename	NET:MANAGE
list	one.vnpool.info	NET:USE
lock	one.vn.lock	NET:MANAGE
unlock	one.vn.unlock	NET:MANAGE

oneuser

oneuser command	XML-RPC Method	Auth. Request
create	one.user.allocate	USER:CREATE
delete	one.user.delete	USER:ADMIN
show	one.user.info	USER:USE
passwd	one.user.passwd	USER:MANAGE
login	one.user.login	USER:MANAGE
update	one.user.update	USER:MANAGE
chauth	one.user.chauth	USER:ADMIN
quota	one.user.quota	USER:ADMIN
chgrp	one.user.chgrp	USER:MANAGE GROUP:MANAGE
addgroup	one.user.addgroup	USER:MANAGE GROUP:MANAGE
delgroup	one.user.delgroup	USER:MANAGE GROUP:MANAGE
encode	–	–
list	one.userpool.info	USER:USE
–	one.userquota.info	–
defaultquota	one.userquota.update	Ony for users in the `oneadmin` group

onedatastore

onedatastore command	XML-RPC Method	Auth. Request
create	one.datastore.allocate	DATASTORE:CREATE [CLUSTER:ADMIN]
delete	one.datastore.delete	DATASTORE:ADMIN
show	one.datastore.info	DATASTORE:USE
update	one.datastore.update	DATASTORE:MANAGE
rename	one.datastore.rename	DATASTORE:MANAGE
chown chgrp	one.datastore.chown	DATASTORE:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.datastore.chmod	DATASTORE:<MANAGE / ADMIN>
enable disable	one.datastore.enable	DATASTORE:MANAGE
list	one.datastorepool.info	DATASTORE:USE

oneimage

oneimage command	XML-RPC Method	Auth. Request
persistent nonpersistent	one.image.persistent	IMAGE:MANAGE
enable disable	one.image.enable	IMAGE:MANAGE
chtype	one.image.chtype	IMAGE:MANAGE
snapshot-delete	one.image.snapshotdelete	IMAGE:MANAGE
snapshot-revert	one.image.snapshotrevert	IMAGE:MANAGE
snapshot-flatten	one.image.snapshotflatten	IMAGE:MANAGE
update	one.image.update	IMAGE:MANAGE
create	one.image.allocate	IMAGE:CREATE DATASTORE:USE
clone	one.image.clone	IMAGE:CREATE IMAGE:USE DATASTORE:USE
delete	one.image.delete	IMAGE:MANAGE
show	one.image.info	IMAGE:USE
chown chgrp	one.image.chown	IMAGE:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.image.chmod	IMAGE:<MANAGE / ADMIN>
rename	one.image.rename	IMAGE:MANAGE
list top	one.imagepool.info	IMAGE:USE
lock	one.image.lock	IMAGE:MANAGE
unlock	one.image.unlock	IMAGE:MANAGE

onemarket

onemarket command	XML-RPC Method	Auth. Request
update	one.market.update	MARKETPLACE:MANAGE
create	one.market.allocate	MARKETPLACE:CREATE
delete	one.market.delete	MARKETPLACE:MANAGE
show	one.market.info	MARKETPLACE:USE
chown chgrp	one.market.chown	MARKETPLACE:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.market.chmod	MARKETPLACE:<MANAGE / ADMIN>
rename	one.market.rename	MARKETPLACE:MANAGE
list	one.marketpool.info	MARKETPLACE:USE

onemarketapp

onemarketapp command	XML-RPC Method	Auth. Request
create	one.marketapp.allocate	MARKETPLACEAPP:CREATE MARKETPLACE:USE
export	– (ruby method)	MARKETPLACEAPP:USE IMAGE:CREATE DATASTORE:USE [TEMPLATE:CREATE]
download	– (ruby method)	MARKETPLACEAPP:USE
enable disable	one.marketapp.enable	MARKETPLACEAPP:MANAGE
update	one.marketapp.update	MARKETPLACEAPP:MANAGE
delete	one.marketapp.delete	MARKETPLACEAPP:MANAGE
show	one.marketapp.info	MARKETPLACEAPP:USE
chown chgrp	one.marketapp.chown	MARKETPLACEAPP:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.marketapp.chmod	MARKETPLACEAPP:<MANAGE / ADMIN>
rename	one.marketapp.rename	MARKETPLACEAPP:MANAGE
list	one.marketapppool.info	MARKETPLACEAPP:USE
lock	one.marketapp.lock	MARKETPLACEAPP:MANAGE
unlock	one.marketapp.unlock	MARKETPLACEAPP:MANAGE

onevrouter

onevrouter command	XML-RPC Method	Auth. Request
create	one.vrouter.allocate	VROUTER:CREATE
update	one.vrouter.update	VROUTER:MANAGE
instantiate	one.vrouter.instantiate	TEMPLATE:USE [IMAGE:USE] [NET:USE]
nic-attach	one.vrouter.attachnic	VROUTER:MANAGE NET:USE
nic-detach	one.vrouter.detachnic	VROUTER:MANAGE
delete	one.vrouter.delete	VROUTER:MANAGE
show	one.vrouter.info	VROUTER:USE
chown chgrp	one.vrouter.chown	VROUTER:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.vrouter.chmod	VROUTER:<MANAGE/ADMIN>
rename	one.vrouter.rename	VROUTER:MANAGE
list top	one.vrouterpool.info	VROUTER:USE
lock	one.vrouter.lock	VROUTER:MANAGE
unlock	one.vrouter.unlock | VROUTER:MANAGE	

onezone

onezone command	XML-RPC Method	Auth. Request
create	one.zone.allocate	ZONE:CREATE
rename	one.zone.rename	ZONE:MANAGE
update	one.zone.update	ZONE:MANAGE
delete	one.zone.delete	ZONE:ADMIN
show	one.zone.info	ZONE:USE
list	one.zonepool.info	ZONE:USE
set	–	ZONE:USE

onesecgroup

onesecgroup command	XML-RPC Method	Auth. Request
create	one.secgroup.allocate	SECGROUP:CREATE
clone	one.secgroup.clone	SECGROUP:CREATE SECGROUP:USE
delete	one.secgroup.delete	SECGROUP:MANAGE
chown chgrp	one.secgroup.chown	SECGROUP:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.secgroup.chmod	SECGROUP:<MANAGE / ADMIN>
update	one.secgroup.update	SECGROUP:MANAGE
commit	one.secgroup.commit	SECGROUP:MANAGE
rename	one.secgroup.rename	SECGROUP:MANAGE
show	one.secgroup.info	SECGROUP:USE
list	one.secgrouppool.info	SECGROUP:USE

onevmgroup

onevmgroup command	XML-RPC Method	Auth. Request
create	one.vmgroup.allocate	VMGROUP:CREATE
delete	one.vmgroup.delete	VMGROUP:MANAGE
chown chgrp	one.vmgroup.chown	VMGROUP:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.vmgroup.chmod	VMGROUP:<MANAGE / ADMIN>
update	one.vmgroup.update	VMGROUP:MANAGE
rename	one.vmgroup.rename	VMGROUP:MANAGE
show	one.vmgroup.info	VMGROUP:USE
list	one.vmgrouppool.info	VMGROUP:USE
lock	one.vmgroup.lock	VMGROUP:MANAGE
unlock	one.vmgroup.unlock	VMGROUP:MANAGE

oneacl

oneacl command	XML-RPC Method	Auth. Request
create	one.acl.addrule	ACL:MANAGE
delete	one.acl.delrule	ACL:MANAGE
list	one.acl.info	ACL:MANAGE

oneacct

command	XML-RPC Method	Auth. Request
oneacct	one.vmpool.accounting	VM:USE

oneshowback

command	XML-RPC Method	Auth. Request
list	one.vmpool.showback	VM:USE
calculate	one.vmpool.calculateshowback	Only for oneadmin group

documents

XML-RPC Method	Auth. Request
one.document.update	DOCUMENT:MANAGE
one.document.allocate	DOCUMENT:CREATE
one.document.clone	DOCUMENT:CREATE DOCUMENT:USE
one.document.delete	DOCUMENT:MANAGE
one.document.info	DOCUMENT:USE
one.document.chown	DOCUMENT:MANAGE [USER:MANAGE] [GROUP:USE]
one.document.chmod	DOCUMENT:<MANAGE / ADMIN>
one.document.rename	DOCUMENT:MANAGE
one.document.lock	DOCUMENT:MANAGE
one.document.unlock	DOCUMENT:MANAGE
one.documentpool.info	DOCUMENT:USE
one.document.lock	DOCUMENT:MANAGE
one.document.unlock	DOCUMENT:MANAGE

system

command	XML-RPC Method	Auth. Request
–	one.system.version	–
–	one.system.config	Ony for users in the `oneadmin` group

onevntemplate

onevntemplate command	XML-RPC Method	Auth. Request
update	one.vntemplate.update	VNTEMPLATE:MANAGE
instantiate	one.vntemplate.instantiate	VNTEMPLATE:USE
create	one.vntemplate.allocate	VNTEMPLATE:CREATE
clone	one.vntemplate.clone	VNTEMPLATE:CREATE VNTEMPLATE:USE
delete	one.vntemplate.delete	VNTEMPLATE:MANAGE
show	one.vntemplate.info	VNTEMPLATE:USE
chown chgrp	one.vntemplate.chown	VNTEMPLATE:MANAGE [USER:MANAGE] [GROUP:USE]
chmod	one.vntemplate.chmod	VNTEMPLATE:<MANAGE/ADMIN>
rename	one.vntemplate.rename	VNTEMPLATE:MANAGE
list top	one.vntemplatepool.info	VNTEMPLATE:USE
lock	one.vntemplate.lock	VNTEMPLATE:MANAGE
unlock	one.vntemplate.unlock	VNTEMPLATE:MANAGE

onehook

onevntemplate command	XML-RPC Method	Auth. Request
update	one.hook.update	HOOK:MANAGE
create	one.hook.allocate	HOOK:CREATE
delete	one.hook.delete	HOOK:MANAGE
show	one.hook.info	HOOK:USE
rename	one.hook.rename	HOOK:MANAGE
list top	one.hook.info	HOOK:USE
lock	one.hook.lock	HOOK:MANAGE
unlock	one.hook.unlock	HOOK:MANAGE
retry	one.hook.unlock	HOOK:MANAGE
log	one.hooklog.info	HOOK:-

1.2.2 Actions for Templates Management

one.template.allocate

- **Description**: Allocates a new template in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.

one.template.clone

- **Description**: Clones an existing virtual machine template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The ID of the template to be cloned.
IN	String	Name for the new template.
IN	Boolean	true to clone the template plus any image defined in DISK. The new IMAGE_ID is set into each DISK.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new template ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the original object that caused the error.

one.template.delete

- **Description**: Deletes the given template from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	true to delete the template plus any image defined in DISK.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.template.instantiate

- **Description**: Instantiates a new virtual machine from a template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	Name for the new VM instance. If it is an empty string, OpenNebula will assign one automatically.
IN	Boolean	False to create the VM on pending (default), True to create it on hold.
IN	String	A string containing an extra template to be merged with the one being instantiated. It can be empty. Syntax can be the usual `attribute=value` or XML.
IN	Boolean	true to create a private persistent copy of the template plus any image defined in DISK, and instantiate that copy.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new virtual machine ID / The error string.
OUT	Int	Error code.

Sample template string:

```
MEMORY=4096\nCPU=4\nVCPU=4
```

Note: Declaring a field overwrites the template. Thus, declaring `DISK=[...]` overwrites the template `DISK` attribute and as such, must contain the entire `DISK` definition.

one.template.update

- **Description**: Replaces the template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.template.chmod

- **Description**: Changes the permission bits of a template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
IN	Boolean	true to chmod the template plus any image defined in DISK.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.template.chown

- **Description**: Changes the ownership of a template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.template.rename

- **Description**: Renames a template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.template.info

- **Description**: Retrieves information for the template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to process the template and include extended information, such as the SIZE for each DISK
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.templatepool.info

- **Description**: Retrieves information for all or part of the Resources in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool,

use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

one.template.lock

- **Description**: Locks a Template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.template.unlock

- **Description**: Unlocks a Template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

1.2.3 Actions for Virtual Machine Management

one.vm.allocate

- **Description**: Allocates a new virtual machine in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template for the vm. Syntax can be the usual `attribute=value` or XML.
IN	Boolean	False to create the VM on pending (default), True to create it on hold.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.vm.deploy

- **Description**: initiates the instance of the given vmid on the target host.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The Host ID of the target host where the VM will be deployed.
IN	Boolean	true to enforce the Host capacity is not overcommitted.
IN	Int	The Datastore ID of the target system datastore where the VM will be deployed. It is optional, and can be set to -1 to let OpenNebula choose the datastore.
IN	String	Template with network scheduling results for NIC in AUTO mode.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Datastore that caused the error.

one.vm.action

- **Description**: submits an action to be performed on a virtual machine.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	the action name to be performed, see below.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

The action String must be one of the following:

- **terminate-hard**
- **terminate**
- **undeploy-hard**
- **undeploy**
- **poweroff-hard**
- **poweroff**
- **reboot-hard**
- **reboot**
- **hold**
- **release**
- **stop**
- **suspend**

- **resume**
- **resched**
- **unresched**

one.vm.migrate

- **Description**: migrates one virtual machine (vid) to the target host (hid).
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	the target host id (hid) where we want to migrate the vm.
IN	Boolean	if true we are indicating that we want livemigration, otherwise false.
IN	Boolean	true to enforce the Host capacity is not overcommitted.
IN	Int	the target system DS id where we want to migrate the vm.
IN	Int	The migration type (0 save, 1 poweroff, 2 poweroff hard).
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Datastore / Host that caused the error.

one.vm.disksaveas

- **Description**: Sets the disk to be saved in the given image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Disk ID of the disk we want to save.
IN	String	Name for the new Image where the disk will be saved.
IN	String	Type for the new Image. If it is an empty string, then the default one will be used. See the existing types in the Image template reference.
IN	Int	Id of the snapshot to export, if -1 the current image state will be used.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new allocated Image ID / The error string. If the Template was cloned, the new Template ID is not returned. The Template can be found by name: "<image_name>-<image_id>"
OUT	Int	Error code.
OUT	Int	ID of the Image / Datastore that caused the error.

one.vm.disksnapshotcreate

- **Description**: Takes a new snapshot of the disk image
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Disk ID of the disk we want to snpashot.
IN	String	Description for the snapshot.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new snapshot ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Image that caused the error.

one.vm.disksnapshotdelete

- **Description**: Deletes a disk snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Disk ID of the disk we want to delete.
IN	Int	ID of the snapshot to be deleted.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The ID of the snapshot deleted/ The error string.
OUT	Int	Error code.
OUT	Int	ID of the Image that caused the error.

one.vm.disksnapshotrevert

- **Description**: Reverts disk state to a previously taken snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Disk ID of the disk to revert its state.
IN	Int	Snapshot ID to revert the disk state to.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The snapshot ID used / The error string.
OUT	Int	Error code.

one.vm.disksnapshotrename

- **Description**: Renames a disk snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	VM ID.
IN	Int	Disk ID.
IN	Int	Snapshot ID.
IN	String	New snapshot name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

one.vm.attach

- **Description**: Attaches a new disk to the virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	A string containing a single DISK vector attribute. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

Sample DISK vector attribute:

```
DISK=[IMAGE_ID=42, TYPE=RBD, DEV_PREFIX=vd, SIZE=123456, TARGET=vdc]
```

one.vm.detach

- **Description**: Detaches a disk from a virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The disk ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

one.vm.diskresize

- **Description**: Resizes a disk of a virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The disk ID.
IN	String	The new size string.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Image that caused the error.

one.vm.attachnic

- **Description**: Attaches a new network interface to the virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	A string containing a single NIC vector attribute. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Machine that caused the error.

one.vm.detachnic

- **Description**: Detaches a network interface from a virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The nic ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

one.vm.chmod

- **Description**: Changes the permission bits of a virtual machine.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vm.chown

- **Description**: Changes the ownership of a virtual machine.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vm.rename

- **Description**: Renames a virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vm.snapshotcreate

- **Description**: Creates a new virtual machine snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new snapshot name. It can be empty.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new snapshot ID / The error string.
OUT	Int	Error code.

one.vm.snapshotrevert

- **Description**: Reverts a virtual machine to a snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The snapshot ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

one.vm.snapshotdelete

- **Description**: Deletes a virtual machine snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The snapshot ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

one.vm.resize

- **Description**: Changes the capacity of the virtual machine
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	Template containing the new capacity elements CPU, VCPU, MEMORY. If one of them is not present, or its value is 0, it will not be resized.
IN	Boolean	true to enforce the Host capacity is not overcommitted. This parameter is only acknoledged for users in the oneadmin group, Host capacity will be always enforced for regular users.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Machine / Host that caused the error.

one.vm.update

- **Description**: Replaces the **user template** contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new user template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vm.updateconf

- **Description**: Updates (appends) a set of supported configuration attributes in the VM template
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Machine that caused the error.

The supported attributes are:

Attribute	Sub-attributes
OS	ARCH, MACHINE, KERNEL, INITRD, BOOTLOADER, BOOT
FEATURES	ACPI, PAE, APIC, LOCALTIME, HYPERV, GUEST_AGENT
INPUT	TYPE, BUS
GRAPHICS	TYPE, LISTEN, PASSWD, KEYMAP
RAW	DATA, DATA_VMX, TYPE
CONTEXT	Any value. **Variable substitution will be made**

Note: Visit the Virtual Machine Template reference for a complete description of each attribute

one.vm.recover

- **Description**: Recovers a stuck VM that is waiting for a driver operation. The recovery may be done by failing or succeeding the pending operation. You need to manually check the vm status on the host, to decide if the operation was successful or not.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Recover operation: success (1), failure (0), retry (2), delete (3), delete-recreate (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Machine that caused the error.

one.vm.info

- **Description**: Retrieves information for the virtual machine.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VM / DS / Cluster / Host that caused the error.

one.vm.monitoring

- **Description**: Returns the virtual machine monitoring records.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The monitoring information string / The error string.
OUT	Int	Error code.

The monitoring information returned is a list of VM elements. Each VM element contains the complete xml of the VM with the updated information returned by the poll action.

For example:

```
<MONITORING_DATA>
    <VM>
        ...
        <LAST_POLL>123</LAST_POLL>
        ...
    </VM>
    <VM>
        ...
        <LAST_POLL>456</LAST_POLL>
        ...
    </VM>
</MONITORING_DATA>
```

one.vm.lock

- **Description**: Locks a Virtual Machine. Lock certain actions depending on blocking level.
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vm.unlock

- **Description**: Unlocks a Virtual Machine.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vmpool.info

- **Description**: Retrieves information for all or part of the VMs in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
IN	Int	VM state to filter by.
IN	String	Filter in KEY=VALUE format.
OUT	Boolean	true or false whenever is successful or not
OUT	String	Version of the VM Pool with a short VM body documents.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

The state filter can be one of the following:

Value	State
-2	Any state, including DONE
-1	Any state, except DONE
0	INIT
1	PENDING
2	HOLD
3	ACTIVE
4	STOPPED
5	SUSPENDED
6	DONE
8	POWEROFF
9	UNDEPLOYED
10	CLONING
11	CLONING_FAILURE

Warning: Value 7 is reserved for FAILED VMs for compatibility reasons.

one.vmpool.infoextended

- **Description**: Retrieves information for all or part of the VMs in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
IN	Int	VM state to filter by.
IN	String	Filter in KEY=VALUE format.
OUT	Boolean	true or false whenever is successful or not
OUT	String	Version of the VM Pool with a short VM body documents.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

The state filter can be one of the following:

Value	State
-2	Any state, including DONE
-1	Any state, except DONE
0	INIT
1	PENDING
2	HOLD
3	ACTIVE
4	STOPPED
5	SUSPENDED
6	DONE
8	POWEROFF
9	UNDEPLOYED
10	CLONING
11	CLONING_FAILURE

Warning: Value 7 is reserved for FAILED VMs for compatibility reasons.

one.vmpool.monitoring

- **Description**: Returns all the virtual machine monitoring records.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.

See *one.vm.monitoring*.

Sample output:

```
<MONITORING_DATA>
    <VM>
        <ID>0</ID>
        <LAST_POLL>123</LAST_POLL>
        ...
    </VM>
    <VM>
        <ID>0</ID>
        <LAST_POLL>456</LAST_POLL>
        ...
    </VM>
    <VM>
        <ID>3</ID>
        <LAST_POLL>123</LAST_POLL>
        ...
    </VM>
    <VM>
        <ID>3</ID>
        <LAST_POLL>456</LAST_POLL>
        ...
    </VM>
</MONITORING_DATA>
```

one.vmpool.accounting

- **Description**: Returns the virtual machine history records.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	Start time for the time interval. Can be -1, in which case the time interval won't have a left boundary.
IN	Int	End time for the time interval. Can be -1, in which case the time interval won't have a right boundary.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.

The XML output is explained in detail in the ''oneacct" guide.

one.vmpool.showback

- **Description**: Returns the virtual machine showback records
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag **- < = -3**: Connected user's resources **- -2**: All resources **- -1**: Connected user's and his group's resources **- > = 0**: UID User's Resources
IN	Int	First month for the time interval. January is 1. Can be -1, in which case the time interval won't have a left boundary.
IN	Int	First year for the time interval. Can be -1, in which case the time interval won't have a left boundary.
IN	Int	Last month for the time interval. January is 1. Can be -1, in which case the time interval won't have a right boundary.
IN	Int	Last year for the time interval. Can be -1, in which case the time interval won't have a right boundary.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.

The XML output will be similar to this one:

```
<SHOWBACK_RECORDS>

  <SHOWBACK>
    <VMID>4315</VMID>
    <VMNAME>vm_4315</VMNAME>
    <UID>2467</UID>
    <GID>102</GID>
    <UNAME>cloud_user</UNAME>
    <GNAME>vdc-test</GNAME>
    <YEAR>2014</YEAR>
    <MONTH>11</MONTH>
    <CPU_COST>13</CPU_COST>
    <MEMORY_COST>21</MEMORY_COST>
    <DISK_COST>7</DISK_COST>
    <TOTAL_COST>41</TOTAL_COST>
    <HOURS>10</HOURS>
  </SHOWBACK>

  <SHOWBACK>
    ...
  </SHOWBACK>

  ...
</SHOWBACK_RECORDS>
```

one.vmpool.calculateshowback

- **Description**: Processes all the history records, and stores the monthly cost for each VM
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	First month for the time interval. January is 1. Can be -1, in which case the time interval won't have a left boundary.
IN	Int	First year for the time interval. Can be -1, in which case the time interval won't have a left boundary.
IN	Int	Last month for the time interval. January is 1. Can be -1, in which case the time interval won't have a right boundary.
IN	Int	Last year for the time interval. Can be -1, in which case the time interval won't have a right boundary.
OUT	Boolean	true or false whenever is successful or not
OUT	String	Empty / The error string.
OUT	Int	Error code.

1.2.4 Actions for Hosts Management

one.host.allocate

- **Description**: Allocates a new host in OpenNebula
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	Hostname of the machine we want to add
IN	String	The name of the information manager (im_mad_name), this values are taken from the oned.conf with the tag name IM_MAD (name)
IN	String	The name of the virtual machine manager mad name (vmm_mad_name), this values are taken from the oned.conf with the tag name VM_MAD (name)
IN	Int	The cluster ID. If it is -1, the default one will be used.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated Host ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.host.delete

- **Description**: Deletes the given host from the pool
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.host.status

- **Description**: Sets the status of the host
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The Host ID.
IN	Int	0: ENABLED 1: DISABLED 2: OFFLINE
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the host that caused the error.

one.host.update

- **Description**: Replaces the host's template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.host.rename

- **Description**: Renames a host.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.host.info

- **Description**: Retrieves information for the host.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.host.monitoring

- **Description**: Returns the host monitoring records.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The monitoring information string / The error string.
OUT	Int	Error code.

The monitoring information returned is a list of HOST elements. Each HOST element contains the complete xml of the host with the updated information returned by the poll action.

For example:

```
<MONITORING_DATA>
    <HOST>
        ...
        <LAST_MON_TIME>123</LAST_MON_TIME>
        ...
    </HOST>
    <HOST>
        ...
        <LAST_MON_TIME>456</LAST_MON_TIME>
        ...
    </HOST>
</MONITORING_DATA>
```

one.hostpool.info

- **Description**: Retrieves information for all the hosts in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.hostpool.monitoring

- **Description**: Returns all the host monitoring records.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.

Sample output:

```
<MONITORING_DATA>
    <HOST>
        <ID>0</ID>
        <LAST_MON_TIME>123</LAST_MON_TIME>
        ...
    </HOST>
    <HOST>
        <ID>0</ID>
        <LAST_MON_TIME>456</LAST_MON_TIME>
        ...
    </HOST>
    <HOST>
        <ID>3</ID>
        <LAST_MON_TIME>123</LAST_MON_TIME>
        ...
    </HOST>
    <HOST>
        <ID>3</ID>
        <LAST_MON_TIME>456</LAST_MON_TIME>
        ...
    </HOST>
</MONITORING_DATA>
```

1.2.5 Actions for Cluster Management

one.cluster.allocate

- **Description**: Allocates a new cluster in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	Name for the new cluster.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated cluster ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.delete

- **Description**: Deletes the given cluster from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.cluster.update

- **Description**: Replaces the cluster template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Object that caused the error.

one.cluster.addhost

- **Description**: Adds a host to the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The host ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster/host that caused the error.

one.cluster.delhost

- **Description**: Removes a host from the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The host ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster/host that caused the error.

one.cluster.adddatastore

- **Description**: Adds a datastore to the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The datastore ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.deldatastore

- **Description**: Removes a datastore from the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The datastore ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.addvnet

- **Description**: Adds a vnet to the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The vnet ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.delvnet

- **Description**: Removes a vnet from the given cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The cluster ID.
IN	Int	The vnet ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.rename

- **Description**: Renames a cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.cluster.info

- **Description**: Retrieves information for the cluster.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.clusterpool.info

- **Description**: Retrieves information for all the clusters in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

1.2.6 Actions for Virtual Network Management

one.vn.allocate

- **Description**: Allocates a new virtual network in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the virtual network. Syntax can be the usual `attribute=value` or XML.
IN	Int	The cluster ID. If it is -1, the default one will be used.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.vn.delete

- **Description**: Deletes the given virtual network from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vn.add_ar

- **Description**: Adds address ranges to a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	template of the address ranges to add. Syntax can be the usual `attribute=value` or XML, see below.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Network that caused the error.

Examples of valid templates:

```
AR = [
    TYPE = IP4,
    IP = 192.168.0.5,
    SIZE = 10 ]
```

```
<TEMPLATE>
  <AR>
    <TYPE>IP4</TYPE>
    <IP>192.168.0.5</IP>
    <SIZE>10</SIZE>
  </AR>
</TEMPLATE>
```

one.vn.rm_ar

- **Description**: Removes an address range from a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	ID of the address range to remove.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Network that caused the error.

one.vn.update_ar

- **Description**: Updates the attributes of an address range.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	template of the address ranges to update. Syntax can be the usual `attribute=value` or XML, see below.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Network that caused the error.

Examples of valid templates:

```
AR = [
    AR_ID = 7,
    GATEWAY = "192.168.30.2",
    EXTRA_ATT = "VALUE",
    SIZE = 10 ]
```

```
<TEMPLATE>
  <AR>
    <AR_ID>7</AR_ID>
    <GATEWAY>192.168.30.2</GATEWAY>
    <EXTRA_ATT>VALUE</EXTRA_ATT>
    <SIZE>10</SIZE>
  </AR>
</TEMPLATE>
```

one.vn.reserve

- **Description**: Reserve network addresses.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The virtual network to reserve from.
IN	String	Template, see below.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Network that caused the error.

The third parameter must be an OpenNebula ATTRIBUTE=VALUE template, with these values:

Attribute	Description	Mandatory
SIZE	Size of the reservation	YES
NAME	If set, the reservation will be created in a new Virtual Network with this name	NO
AR_ID	ID of the AR from where to take the addresses	NO
NETWORK_ID	Instead of creating a new Virtual Network, the reservation will be added to the existing virtual network with this ID.	NO
MAC	First MAC address to start the reservation range [MAC, MAC+SIZE)	NO
IP	First IPv4 address to start the reservation range [IP, IP+SIZE)	NO

one.vn.free_ar

- **Description**: Frees a reserved address range from a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	ID of the address range to free.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VNet that caused the error.

one.vn.hold

- **Description**: Holds a virtual network Lease as used.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	template of the lease to hold, e.g. `LEASES=[IP=192.168.0.5]`.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VNet that caused the error.

one.vn.release

- **Description**: Releases a virtual network Lease on hold.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	template of the lease to release, e.g. `LEASES=[IP=192.168.0.5]`.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VNet that caused the error.

one.vn.update

- **Description**: Replaces the virtual network template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VNet that caused the error.

one.vn.chmod

- **Description**: Changes the permission bits of a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vn.chown

- **Description**: Changes the ownership of a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vn.rename

- **Description**: Renames a virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vn.info

- **Description**: Retrieves information for the virtual network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

Note: The ACL rules do not apply to VNET reserveations in the same way as they do to normal VNETs and other objects. Read more in the ACL documentation guide.

one.vn.lock

- **Description**: Locks a Virtual Network. Lock certain actions depending on blocking level.
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the Object that caused the error.	

one.vn.unlock

- **Description**: Unlocks a Virtual Network.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the Object that caused the error.	

one.vnpool.info

- **Description**: Retrieves information for all or part of the virtual networks in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

Note: The ACL rules do not apply to VNET reserveations in the same way as they do to normal VNETs and other objects. Read more in the ACL documentation guide.

1.2.7 Actions for Security Group Management

one.secgroup.allocate

- **Description**: Allocates a new security group in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the security group. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.secgroup.clone

- **Description**: Clones an existing security group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The ID of the security group to be cloned.
IN	String	Name for the new security group.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new security group ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the original object that caused the error.

one.secgroup.delete

- **Description**: Deletes the given security group from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgroup.update

- **Description**: Replaces the security group template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.secgroup.commit

- **Description**: Commit security group changes to associated VMs. This is intended for retrying updates of VMs or reinitialize the updating process if oned stopped or failed after a one.secgroup.update call.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	I true the action will only operate on outdated and error VMs. False to update all VMs.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgroup.chmod

- **Description**: Changes the permission bits of a security group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgroup.chown

- **Description**: Changes the ownership of a security group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgroup.rename

- **Description**: Renames a security group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgroup.info

- **Description**: Retrieves information for the security group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.secgrouppool.info

- **Description**: Retrieves information for all or part of the security groups in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.8 Actions for VM Group Management

one.vmgroup.allocate

- **Description**: Allocates a new VM group in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the VM. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.vmgroup.delete

- **Description**: Deletes the given VM group from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vmgroup.update

- **Description**: Replaces the VM group template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vmgroup.chmod

- **Description**: Changes the permission bits of a VM group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vmgroup.chown

- **Description**: Changes the ownership of a VM group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vmgroup.rename

- **Description**: Renames a VM group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vmgroup.info

- **Description**: Retrieves information for the VM group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vmgroup.lock

- **Description**: Locks a Virtual Machine Group. Lock certain actions depending on blocking level.
- **USE**: locks Admin, Manage and Use actions.

- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vmgroup.unlock

- **Description**: Unlocks a Virtual Machine Group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vmgrouppool.info

- **Description**: Retrieves information for all or part of the VM groups in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.9 Actions for Datastore Management

one.datastore.allocate

- **Description**: Allocates a new datastore in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the datastore. Syntax can be the usual `attribute=value` or XML.
IN	Int	The cluster ID. If it is -1, the default one will be used.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.datastore.delete

- **Description**: Deletes the given datastore from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.datastore.update

- **Description**: Replaces the datastore template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.datastore.chmod

- **Description**: Changes the permission bits of a datastore.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.datastore.chown

- **Description**: Changes the ownership of a datastore.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.datastore.rename

- **Description**: Renames a datastore.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.datastore.enable

- **Description**: Enables or disables a datastore.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	True for enabling, false for disabling.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Datastore that caused the error.

one.datastore.info

- **Description**: Retrieves information for the datastore.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.datastorepool.info

- **Description**: Retrieves information for all or part of the datastores in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

1.2.10 Actions for Image Management

one.image.allocate

- **Description**: Allocates a new image in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the image. Syntax can be the usual `attribute=value` or XML.
IN	Int	The datastore ID.
IN	Boolean	true to avoid checking datastore capacity. Only for admins.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Datastore that caused the error.

one.image.clone

- **Description**: Clones an existing image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The ID of the image to be cloned.
IN	String	Name for the new image.
IN	Int	The ID of the target datastore. Optional, can be set to -1 to use the current one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new image ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the original Image / DS or destination DS that caused the error.

one.image.delete

- **Description**: Deletes the given image from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.image.enable

- **Description**: Enables or disables an image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The Image ID.
IN	Boolean	True for enabling, false for disabling.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The Image ID / The error string.
OUT	Int	Error code.

one.image.persistent

- **Description**: Sets the Image as persistent or not persistent.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The Image ID.
IN	Boolean	True for persistent, false for non-persisent.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The Image ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the image that caused the error.

one.image.chtype

- **Description**: Changes the type of an Image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The Image ID.
IN	String	New type for the Image. See the existing types in the Image template reference.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The Image ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the image that caused the error.

one.image.update

- **Description**: Replaces the image template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.image.chmod

- **Description**: Changes the permission bits of an image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.image.chown

- **Description**: Changes the ownership of an image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.image.rename

- **Description**: Renames an image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.image.snapshotdelete

- **Description**: Deletes a snapshot from the image
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	ID of the snapshot to delete
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	ID of the deleted snapshot/The error string.
OUT	Int	Error code.

one.image.snapshotrevert

- **Description**: Reverts image state to a previous snapshot
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	ID of the snapshot to revert to
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	ID of the snapshot/The error string.
OUT	Int	Error code.

one.image.snapshotflatten

- **Description**: Flatten the snapshot of image and discards others
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	ID of the snapshot to flatten
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	ID of the snapshot/The error string.
OUT	Int	Error code.

one.image.info

- **Description**: Retrieves information for the image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.Image.lock

- **Description**: Locks an Image. Lock certain actions depending on blocking level
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.image.unlock

- **Description**: Unlocks an Image.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.imagepool.info

- **Description**: Retrieves information for all or part of the images in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.11 Actions for Marketplace Management

one.market.allocate

- **Description**: Allocates a new marketplace in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the marketplace. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.market.delete

- **Description**: Deletes the given marketplace from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.market.update

- **Description**: Replaces the marketplace template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.market.chmod

- **Description**: Changes the permission bits of a marketplace.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.market.chown

- **Description**: Changes the ownership of a marketplace.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.market.rename

- **Description**: Renames a marketplace.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.market.info

- **Description**: Retrieves information for the marketplace.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketpool.info

- **Description**: Retrieves information for all or part of the marketplaces in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

1.2.12 Actions for MarketplaceApp Management

one.marketapp.allocate

- **Description**: Allocates a new marketplace app in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the marketplace app. Syntax can be the usual `attribute=value` or XML.
IN	Int	The Marketplace ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.marketapp.delete

- **Description**: Deletes the given marketplace app from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketapp.enable

- **Description**: Enables or disables a marketplace app.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The marketplace app ID.
IN	Boolean	True for enabling, false for disabling.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The marketplace app ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the market that caused the error.

one.marketapp.update

- **Description**: Replaces the marketplace app template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.marketapp.chmod

- **Description**: Changes the permission bits of a marketplace app.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketapp.chown

- **Description**: Changes the ownership of a marketplace app.

- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketapp.rename

- **Description**: Renames a marketplace app.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketapp.info

- **Description**: Retrieves information for the marketplace app.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.marketapp.lock

- **Description**: Locks a MarketPlaceApp. Lock certain actions depending on blocking level
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.marketapp.unlock

- **Description**: Unlocks a MarketPlaceApp.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.marketapppool.info

- **Description**: Retrieves information for all or part of the marketplace apps in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.13 Actions for Virtual Routers Management

one.vrouter.allocate

- **Description**: Allocates a new virtual router in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the virtual router contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.vrouter.delete

- **Description**: Deletes the given virtual router from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	true to delete the virtual router plus any image defined in DISK.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vrouter.instantiate

- **Description**: Instantiates a new virtual machine from a virtual router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Number of VMs to instantiate.
IN	Int	VM Template id to instantiate.
IN	String	Name for the VM instances. If it is an empty string OpenNebula will set a default name. Wildcard %i can be used.
IN	Boolean	False to create the VM on pending (default), True to create it on hold.
IN	String	A string containing an extra template to be merged with the one being instantiated. It can be empty. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	ID of the Virtual Router that instantiated the VMs / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Router that caused the error.

Sample template string:

```
MEMORY=4096\nCPU=4\nVCPU=4
```

Note: Declaring a field overwrites the template. Thus, declaring `DISK=[...]` overwrites the template `DISK` attribute and as such, must contain the entire `DISK` definition.

one.vrouter.attachnic

- **Description**: Attaches a new network interface to the virtual router and the virtual machines
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	A string containing a single NIC vector attribute. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Virtual Router that caused the error.

one.vrouter.detachnic

- **Description**: Detaches a network interface from the virtual router and the virtual machines
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The nic ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the VRouter that caused the error.

one.vrouter.update

- **Description**: Replaces the template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vrouter.chmod

- **Description**: Changes the permission bits of a virtual router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vrouter.chown

- **Description**: Changes the ownership of a virtual router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vrouter.rename

- **Description**: Renames a virtual router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vrouter.info

- **Description**: Retrieves information for the virtual router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vrouter.lock

- **Description**: Locks a Virtual Router. Lock certain actions depending on blocking level
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vrouter.unlock

- **Description**: Unlocks a Virtual Router.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vrouterpool.info

- **Description**: Retrieves information for all or part of the Resources in the pool.

- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.14 Actions for User Management

one.user.allocate

- **Description**: Allocates a new user in OpenNebula
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	username for the new user
IN	String	password for the new user
IN	String	authentication driver for the new user. If it is an empty string, then the default ('core') is used
IN	Array	array of Group IDs. The first ID will be used as the main group. This array can be empty, in which case the default group will be used
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated User ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Group that caused the error.

one.user.delete

- **Description**: Deletes the given user from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.user.passwd

- **Description**: Changes the password for the given user.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new password
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The User ID / The error string.
OUT	Int	Error code.

one.user.login

- **Description**: Generates or sets a login token.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	The user name to generate the token for
IN	String	The token, if empty oned will generate one
IN	Int	Valid period in seconds; 0 reset the token and -1 for a non-expiring token.
IN	Int	Effective GID to use with this token. To use the current GID and user groups set it to -1
OUT	Boolean	true or false whenever is successful or not
OUT	String	The new token / The error string.
OUT	Int	Error code.

one.user.update

- **Description**: Replaces the user template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.user.chauth

- **Description**: Changes the authentication driver and the password for the given user.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new authentication driver.
IN	String	The new password. If it is an empty string, the password is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The User ID / The error string.
OUT	Int	Error code.

one.user.quota

- **Description**: Sets the user quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new quota template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.user.chgrp

- **Description**: Changes the group of the given user.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The User ID.
IN	Int	The Group ID of the new group.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The User ID / The error string.
OUT	Int	Error code.

one.user.addgroup

- **Description**: Adds the User to a secondary group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The User ID.
IN	Int	The Group ID of the new group.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The User ID / The error string.
OUT	Int	Error code.

one.user.delgroup

- **Description**: Removes the User from a secondary group
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The User ID.
IN	Int	The Group ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The User ID / The error string.
OUT	Int	Error code.

one.user.info

- **Description**: Retrieves information for the user.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID. If it is -1, then the connected user's own info info is returned
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.userpool.info

- **Description**: Retrieves information for all the users in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.userquota.info

- **Description**: Returns the default user quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The quota template contents / The error string.
OUT	Int	Error code.

one.userquota.update

- **Description**: Updates the default user quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	The new quota template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The quota template contents / The error string.
OUT	Int	Error code.

1.2.15 Actions for Group Management

one.group.allocate

- **Description**: Allocates a new group in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	Name for the new group.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated Group ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.group.delete

- **Description**: Deletes the given group from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.group.info

- **Description**: Retrieves information for the group.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID. If it is -1, then the connected user's group info info is returned
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.group.update

- **Description**: Replaces the group template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.group.addadmin

- **Description**: Adds a User to the Group administrators set
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The group ID.
IN	Int	The user ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.group.deladmin

- **Description**: Removes a User from the Group administrators set
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The group ID.
IN	Int	The user ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.group.quota

- **Description**: Sets the group quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new quota template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the group that caused the error.

one.grouppool.info

- **Description**: Retrieves information for all the groups in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.groupquota.info

- **Description**: Returns the default group quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The quota template contents / The error string.
OUT	Int	Error code.

one.groupquota.update

- **Description**: Updates the default group quota limits.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	The new quota template contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The quota template contents / The error string.
OUT	Int	Error code.

1.2.16 Actions for VDC Management

one.vdc.allocate

- **Description**: Allocates a new VDC in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the VDC. Syntax can be the usual `attribute=value` or XML.
IN	Int	The cluster ID. If it is -1, this virtual network won't be added to any cluster.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.vdc.delete

- **Description**: Deletes the given VDC from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vdc.update

- **Description**: Replaces the VDC template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.rename

- **Description**: Renames a VDC.

- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vdc.info

- **Description**: Retrieves information for the VDC.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID. If it is -1, then the connected user's VDC info info is returned
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vdcpool.info

- **Description**: Retrieves information for all the VDCs in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vdc.addgroup

- **Description**: Adds a group to the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The group ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.delgroup

- **Description**: Deletes a group from the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The group ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.addcluster

- **Description**: Adds a cluster to the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Cluster ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.delcluster

- **Description**: Deletes a cluster from the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Cluster ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.addhost

- **Description**: Adds a host to the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Host ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.delhost

- **Description**: Deletes a host from the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Host ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.adddatastore

- **Description**: Adds a datastore to the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Datastore ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.deldatastore

- **Description**: Deletes a datastore from the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Datastore ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.addvnet

- **Description**: Adds a vnet to the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Vnet ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vdc.delvnet

- **Description**: Deletes a vnet from the VDC
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The VDC ID.
IN	Int	The Zone ID.
IN	Int	The Vnet ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

1.2.17 Actions for Zone Management

one.zone.allocate

- **Description**: Allocates a new zone in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the template of the zone. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

one.zone.delete

- **Description**: Deletes the given zone from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.zone.update

- **Description**: Replaces the zone template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Zone that caused the error.

one.zone.rename

- **Description**: Renames a zone.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.zone.info

- **Description**: Retrieves information for the zone.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.zone.raftstatus

- **Description**: Retrieves raft status one servers.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.zonepool.info

- **Description**: Retrieves information for all the zones in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

1.2.18 Actions for ACL Rules Management

one.acl.addrule

- **Description**: Adds a new ACL rule.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	User component of the new rule. A string containing a hex number.
IN	String	Resource component of the new rule. A string containing a hex number.
IN	String	Rights component of the new rule. A string containing a hex number.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated ACL rule ID / The error string.
OUT	Int	Error code.

To build the hex. numbers required to create a new rule we recommend you to read the ruby or java code.

one.acl.delrule

- **Description**: Deletes an ACL rule.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	ACL rule ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The ACL rule ID / The error string.
OUT	Int	Error code.

one.acl.info

- **Description**: Returns the complete ACL rule set.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

1.2.19 Actions for Document Management

one.document.allocate

- **Description**: Allocates a new document in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the document template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	The document type (*).
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the Cluster that caused the error.

(*) Type is an integer value used to allow dynamic pools compartmentalization.

Let's say you want to store documents representing Chef recipes, and EC2 security groups; you would allocate documents of each kind with a different type. This type is then used in the one.documentpool.info method to filter the results.

one.document.clone

- **Description**: Clones an existing document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The ID of the document to be cloned.
IN	String	Name for the new document.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new document ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the original object that caused the error.

one.document.delete

- **Description**: Deletes the given document from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.document.update

- **Description**: Replaces the document template contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new document template contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: Replace the whole template. **1**: Merge new template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.document.chmod

- **Description**: Changes the permission bits of a document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.document.chown

- **Description**: Changes the ownership of a document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.document.rename

- **Description**: Renames a document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.document.info

- **Description**: Retrieves information for the document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.document.lock

- **Description**: Locks a Document. Lock certain actions depending on blocking level:
- **USE**: locks Admin, Manage and Use actions.
- **MANAGE**: locks Manage and Use actions.
- **ADMIN**: locks only Admin actions.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.document.unlock

- **Description**: Unlocks a Document.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.documentpool.info

- **Description**: Retrieves information for all or part of the Resources in the pool.

- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
IN	Int	The document type.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.20 System Methods

one.system.version

- **Description**: Returns the OpenNebula core version
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The OpenNebula version, e.g. `4.4.0`
OUT	Int	Error code.

one.system.config

- **Description**: Returns the OpenNebula configuration
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The loaded oned.conf file, in XML form
OUT	Int	Error code.

1.2.21 Actions for Virtual Network Templates Management

one.vntemplate.allocate

- **Description**: Allocates a new vntemplate in OpenNebula.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the vntemplate contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.

one.vntemplate.clone

- **Description**: Clones an existing virtual network template.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The ID of the vntemplate to be cloned.
IN	String	Name for the new vntemplate.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new vntemplate ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the original object that caused the error.

one.vntemplate.delete

- **Description**: Deletes the given vntemplate from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vntemplate.instantiate

- **Description**: Instantiates a new virtual network from a vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	Name for the new Virtual Network. If it is an empty string, OpenNebula will assign one automatically.
IN	String	A string containing an extra vntemplate to be merged with the one being instantiated. It can be empty. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The new virtual machine ID / The error string.
OUT	Int	Error code.

Sample vntemplate string:

```
VN_MAD=bridge\nVLAN_ID=4
```

Note: Declaring a field overwrites the vntemplate. Thus, declaring `VN_MAD=[...]` overwrites the vntemplate `VN_MAD` attribute.

one.vntemplate.update

- **Description**: Replaces the vntemplate contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new vntemplate contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: replace the whole vntemplate. **1**: Merge new vntemplate with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.vntemplate.chmod

- **Description**: Changes the permission bits of a vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	USER USE bit. If set to -1, it will not change.
IN	Int	USER MANAGE bit. If set to -1, it will not change.
IN	Int	USER ADMIN bit. If set to -1, it will not change.
IN	Int	GROUP USE bit. If set to -1, it will not change.
IN	Int	GROUP MANAGE bit. If set to -1, it will not change.
IN	Int	GROUP ADMIN bit. If set to -1, it will not change.
IN	Int	OTHER USE bit. If set to -1, it will not change.
IN	Int	OTHER MANAGE bit. If set to -1, it will not change.
IN	Int	OTHER ADMIN bit. If set to -1, it will not change.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vntemplate.chown

- **Description**: Changes the ownership of a vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The User ID of the new owner. If set to -1, the owner is not changed.
IN	Int	The Group ID of the new group. If set to -1, the group is not changed.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vntemplate.rename

- **Description**: Renames a vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vntemplate.info

- **Description**: Retrieves information for the vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.vntemplate.lock

- **Description**: Locks a vntemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vntemplate.unlock

- **Description**: Unlocks a vnemplate.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.vntemplatepool.info

- **Description**: Retrieves information for all or part of the Resources in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

1.2.22 Actions for Hook Management

one.hook.allocate

- **Description**: Allocates a new Hook in OpenNebula.

- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	String	A string containing the hook contents. Syntax can be the usual `attribute=value` or XML.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The allocated resource ID / The error string.
OUT	Int	Error code.

one.hook.delete

- **Description**: Deletes the given hook from the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.hook.update

- **Description**: Replaces the hook contents.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new hook contents. Syntax can be the usual `attribute=value` or XML.
IN	Int	Update type: **0**: replace the whole hook template. **1**: Merge new hook template with the existing one.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The resource ID / The error string.
OUT	Int	Error code.

one.hook.rename

- **Description**: Renames a hook.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	String	The new name.
OUT	Boolean	true or false whenever is successful or not.
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.hook.info

- **Description**: Retrieves information for the hook.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Boolean	optional flag to decrypt contained secrets, valid only for admin
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

one.hook.lock

- **Description**: Locks a hook.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	Lock level: use (1), manage (2), admin (3), all (4)
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.hook.unlock

- **Description**: Unlocks a hook.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.hook.retry

- **Description**: Retries a hook execution.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	The execution ID.
OUT	Boolean	true or false whenever is successful or not
OUT	Int	The ID of the resource.
OUT	Int	Error code.
OUT	Int | ID of the object that caused the error.	

one.hookpool.info

- **Description**: Retrieves information for all or part of the Resources in the pool.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Filter flag • **-4**: Resources belonging to the user's primary group • **-3**: Resources belonging to the user • **-2**: All resources • **-1**: Resources belonging to the user and any of his groups • **>= 0**: UID User's Resources
IN	Int	When the next parameter is >= -1 this is the Range start ID. Can be -1. For smaller values this is the offset used for pagination.
IN	Int	For values >= -1 this is the Range end ID. Can be -1 to get until the last ID. For values < -1 this is the page size used for pagination.
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

The range can be used to retrieve a subset of the pool, from the 'start' to the 'end' ID. To retrieve the complete pool, use `(-1, -1)`; to retrieve all the pool from a specific ID to the last one, use `(<id>, -1)`, and to retrieve the first elements up to an ID, use `(0, <id>)`.

one.hooklog.info

- **Description**: Retrieves information from the hook execution log.
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	Minimun date for filtering hook execution log records.
IN	Int	Maximum date for filtering hook execution log records.
IN	Int	Hook id for filtering hook execution log records.
IN	Int	Hook execution return code (`-1` ERROR, `0` ALL , `1` SUCCESS).
OUT	Boolean	true or false whenever is successful or not
OUT	String	The information string / The error string.
OUT	Int	Error code.
OUT	Int	ID of the object that caused the error.

1.2.23 XSD Reference

The XML schema files that describe the XML documents returned by the one.*.info methods *can be found here <https://github.com/OpenNebula/one/tree/master/share/doc/xsd>*

1.3 Ruby OpenNebula Cloud API

This page contains the OpenNebula Cloud API Specification for Ruby. It has been designed as a wrapper for the *XML-RPC methods*, with some basic helpers. This means that you should be familiar with the XML-RPC API and the XML formats returned by the OpenNebula core. As stated in the *XML-RPC documentation*, you can download the *XML Schemas (XSD) here*.

1.3.1 API Documentation

You can consult the doc online.

1.3.2 Usage

You can use the Ruby OCA included in the OpenNebula distribution by adding the OpenNebula Ruby library path to the search path:

```
################################################################################
# Environment Configuration
################################################################################
ONE_LOCATION=ENV["ONE_LOCATION"]

if !ONE_LOCATION
    RUBY_LIB_LOCATION="/usr/lib/one/ruby"
else
    RUBY_LIB_LOCATION=ONE_LOCATION+"/lib/ruby"
end

$: << RUBY_LIB_LOCATION

################################################################################
# Required libraries
################################################################################
require 'opennebula'
```

1.3.3 Code Sample: Shutdown all the VMs of the Pool

This is a small code snippet. As you can see, the code flow would be as follows:

- Create a new Client, setting up the authorization string. You only need to create it once.
- Get the VirtualMachine pool that contains the VirtualMachines owned by this User.
- You can perform "actions" over these objects right away, like `myVNet.delete();`. In this example all the VirtualMachines will be shut down.

```ruby
#!/usr/bin/env ruby

################################################################################
# Environment Configuration
################################################################################
ONE_LOCATION=ENV["ONE_LOCATION"]

if !ONE_LOCATION
    RUBY_LIB_LOCATION="/usr/lib/one/ruby"
else
    RUBY_LIB_LOCATION=ONE_LOCATION+"/lib/ruby"
end

$: << RUBY_LIB_LOCATION

################################################################################
# Required libraries
################################################################################
require 'opennebula'

include OpenNebula

# OpenNebula credentials
CREDENTIALS = "oneuser:onepass"
# XML_RPC endpoint where OpenNebula is listening
ENDPOINT    = "http://localhost:2633/RPC2"

client = Client.new(CREDENTIALS, ENDPOINT)

vm_pool = VirtualMachinePool.new(client, -1)

rc = vm_pool.info
if OpenNebula.is_error?(rc)
     puts rc.message
     exit -1
end

vm_pool.each do |vm|
     rc = vm.shutdown
     if OpenNebula.is_error?(rc)
          puts "Virtual Machine #{vm.id}: #{rc.message}"
     else
          puts "Virtual Machine #{vm.id}: Shutting down"
     end
end

exit 0
```

1.3.4 Code Sample: Create a new Virtual Network

```ruby
#!/usr/bin/env ruby

################################################################################
# Environment Configuration
################################################################################
```

(continues on next page)

(continued from previous page)

```
ONE_LOCATION=ENV["ONE_LOCATION"]

if !ONE_LOCATION
    RUBY_LIB_LOCATION="/usr/lib/one/ruby"
else
    RUBY_LIB_LOCATION=ONE_LOCATION+"/lib/ruby"
end

$: << RUBY_LIB_LOCATION

################################################################################
# Required libraries
################################################################################
require 'opennebula'

include OpenNebula

# OpenNebula credentials
CREDENTIALS = "oneuser:onepass"
# XML_RPC endpoint where OpenNebula is listening
ENDPOINT    = "http://localhost:2633/RPC2"

client = Client.new(CREDENTIALS, ENDPOINT)

template = <<-EOT
NAME    = "Red LAN"

# Now we'll use the host private network (physical)
BRIDGE  = vbr0

# Custom Attributes to be used in Context
GATEWAY = 192.168.0.1
DNS     = 192.168.0.1

LOAD_BALANCER = 192.168.0.3

AR = [
    TYPE = IP4,
    IP   = 192.168.0.1,
    SIZE = 255
]
EOT

xml = OpenNebula::VirtualNetwork.build_xml
vn  = OpenNebula::VirtualNetwork.new(xml, client)

rc = vn.allocate(template)
if OpenNebula.is_error?(rc)
    STDERR.puts rc.message
    exit(-1)
else
    puts "ID: #{vn.id.to_s}"
end

puts "Before info:"
puts vn.to_xml
```

(continues on next page)

(continued from previous page)

```
puts

vn.info

puts "After info:"
puts vn.to_xml
```

1.4 PyONE: Open Nebula Python Bindings

PyONE is an implementation of Open Nebula XML-RPC bindings in Python. It has been designed as a wrapper for the *XML-RPC methods*, with some basic helpers. This means that you should be familiar with the XML-RPC API and the XML formats returned by the OpenNebula core. As stated in the *XML-RPC documentation*, you can download the *XML Schemas (XSD) here*.

1.4.1 API Documentation

You can consult the doc online, but as long as the code is generated it is not much useful, the main source of the documentation is still the *XML-RPC doc <api>*

1.4.2 Download and installation

You can either use the system package `python-pyone` / `python3-pyone` or install it using `pip install pyone`.

1.4.3 Usage

You can configure your XML-RPC Server endpoint and credentials when instantiate the OneServer class:

```
import pyone
one = pyone.OneServer("http://one:2633/RPC2", session="oneadmin:onepass" )
```

If you are connecting to a test platform with a self signed certificate you can disable certificate verification as:

```
import pyone
import ssl
one = pyone.OneServer("http://one:2633/RPC2", session="oneadmin:onepass", context=ssl.
↪_create_unverified_context() )
```

It is also possible to modify the default connection timeout, but note that the setting will modify the TCP socket default timeout of your Python VM, ensure that the chosen timeout is suitable to any other connections running in your project.

Making Calls

Calls match the API documentation provided by OpenNebula:

```
import pyone

one = pyone.OneServer("http://one:2633/RPC2", session="oneadmin:onepass" )
hostpool = one.hostpool.info()
```

(continues on next page)

(continued from previous page)

```
host = hostpool.HOST[0]
id = host.ID
```

Note that the session parameter is automatically included as well as the "one." prefix to the method.

Returned Objects

The returned types have been generated with generateDS and closely match the XSD specification. You can use the XSD specification and XML-RPC as priamry documentation.

```
marketpool = one.marketpool.info()
m0 = marketpool.MARKETPLACE[0]
print "Markeplace name is " + m0.NAME
```

Structured Parameters

When making calls, the library will translate flat dictionaries into attribute=value vectors. Such as:

```
one.host.update(0,  {"LABELS": "HD"}, 1)
```

When the provided dictionary has a "root" dictionary, it is considered to be root element and it will be translated to XML:

```
one.vm.update(1,
  {
    'TEMPLATE': {
      'NAME': 'abc',
      'MEMORY': '1024',
      'ATT1': 'value1'
    }
  }, 1)
```

However, this might be limiting when you want to add 2 entries with same name. In such cases you need to pass the template directly in OpenNebula template format:

```
one.template.allocate(
  '''NAME="test100"
     MEMORY="1024"
     DISK=[ IMAGE_ID= "1" ]
     DISK=[ IMAGE_ID= "2" ]
     CPU="1"
     VCPU="2"
  ''')
```

generateDS creates members from most returned parameters, however, some elements in the XSD are marked as anyType and generateDS cannot generate members automatically, TEMPLATE and USER_TEMPLATE are the common ones. Pyone will allow accessing its contents as a plain python dictionary.

```
host = one.host.info(0)
arch = host.TEMPLATE['ARCH']
```

This makes it possible to read a TEMPLATE as dictionary, modify it and use it as parameter for an update method, as following:

```
host = one.host.info(0)
host.TEMPLATE['NOTES']="Just updated"
one.host.update(0,host.TEMPLATE,1)
```

Constants

Some methods will return encoded values such as those representing the STATE of a resource. Constant are provided to better handle those.

```
from pyone import MARKETPLACEAPP_STATES
if app.STATE == MARKETPLACEAPP_STATES.READY:
  # action that assumes app ready
```

Exapmles

```
import pyone
one = pyone.OneServer("http://one:2633/RPC2", session="oneadmin:onepass" )
```

Allocate localhost as new host

```
one.host.allocate('localhost', 'kvm', 'kvm', 0)
```

See host template

```
host = one.hostpool.info().HOST[0]
dict(host.TEMPLATE)
```

See VM template

```
vm_template = one.templatepool.info(-1, -1, -1).VMTEMPLATE[0]
vm_template.get_ID()
vm_template.get_NAME()
```

Instantiate it

```
one.template.instantiate(0, "my_VM")
```

See it

```
my_vm = one.vmpool.info(-1,-1,-1,-1).VM[0]
my_vm.get_ID()
my_vm.get_NAME()
my_vm.get_TEMPLATE()
```

Terminate it

```
one.vm.action('terminate', 0)
```

1.4.4 Credits

Python bindings were ported to upstream from stand-alone PyONE addon made by *Rafael del Valle PyONE <https://github.com/OpenNebula/addon-pyone>*

1.5 Java OpenNebula Cloud API

This page contains the OpenNebula Cloud API Specification for Java. It has been designed as a wrapper for the *XML-RPC methods*, with some basic helpers. This means that you should be familiar with the XML-RPC API and the XML formats returned by the OpenNebula core. As stated in the *XML-RPC documentation*, you can download the *XML Schemas (XSD) here*.

1.5.1 Download

You can download the `.jar` file compiled using Java 1.8, the required libraries, and the javadoc packaged in a tar.gz file following this link in the OpenNebula version you have installed.

You can also consult the javadoc online.

1.5.2 Usage

To use the OpenNebula Cloud API for Java in your Java project, you have to add to the classpath the org.opennebula.client.jar file and the xml-rpc libraries located in the lib directory.

1.5.3 Code Sample

This is a small code snippet. As you can see, the code flow would be as follows:

- Create a `org.opennebula.client.Client` object, setting up the authorization string and the endpoint. You only need to create it once.
- Create a pool (e.g. `HostPool`) or element (e.g. `VirtualNetwork`) object.
- You can perform "actions" over these objects right away, like `myVNet.delete();`
- If you want to query any information (like what objects the pool contains, or one of the element attributes), you have to issue an `info()` call before, so the object retrieves the data from OpenNebula.

For more complete examples, please check the `src/oca/java/share/examples` directory included. You may be also interested in the java files included in `src/oca/java/test`.

```
// First of all, a Client object has to be created.
// Here the client will try to connect to OpenNebula using the default
// options: the auth. file will be assumed to be at $ONE_AUTH, and the
// endpoint will be set to the environment variable $ONE_XMLRPC.
Client oneClient;

try
{
    oneClient = new Client();

    // We will try to create a new virtual machine. The first thing we
    // need is an OpenNebula virtual machine template.

    // This VM template is a valid one, but it will probably fail to run
    // if we try to deploy it; the path for the image is unlikely to
    // exist.
    String vmTemplate =
          "NAME     = vm_from_java    CPU = 0.1    MEMORY = 64\n"
        + "#DISK     = [\n"
        + "#\tsource   = \"/home/user/vmachines/ttylinux/ttylinux.img\",\n"
        + "#\ttarget   = \"hda\",\n"
        + "#\treadonly = \"no\" ]\n"
        + "# NIC     = [ NETWORK = \"Non existing network\" ]\n"
        + "FEATURES = [ acpi=\"no\" ]";

    // You can try to uncomment the NIC line, in that case OpenNebula
    // won't be able to insert this machine in the database.
```

(continues on next page)

(continued from previous page)

```
    System.out.println("Virtual Machine Template:\n" + vmTemplate);
    System.out.println();

    System.out.print("Trying to allocate the virtual machine... ");
    OneResponse rc = VirtualMachine.allocate(oneClient, vmTemplate);

    if( rc.isError() )
    {
        System.out.println( "failed!");
        throw new Exception( rc.getErrorMessage() );
    }

    // The response message is the new VM's ID
    int newVMID = Integer.parseInt(rc.getMessage());
    System.out.println("ok, ID " + newVMID + ".");

    // We can create a representation for the new VM, using the returned
    // VM-ID
    VirtualMachine vm = new VirtualMachine(newVMID, oneClient);

    // Let's hold the VM, so the scheduler won't try to deploy it
    System.out.print("Trying to hold the new VM... ");
    rc = vm.hold();

    if(rc.isError())
    {
        System.out.println("failed!");
        throw new Exception( rc.getErrorMessage() );
    }
    else
        System.out.println("ok.");

    // And now we can request its information.
    rc = vm.info();

    if(rc.isError())
        throw new Exception( rc.getErrorMessage() );

    System.out.println();
    System.out.println(
            "This is the information OpenNebula stores for the new VM:");
    System.out.println(rc.getMessage() + "\n");

    // This VirtualMachine object has some helpers, so we can access its
    // attributes easily (remember to load the data first using the info
    // method).
    System.out.println("The new VM " +
            vm.getName() + " has status: " + vm.status());

    // And we can also use xpath expressions
    System.out.println("The path of the disk is");
    System.out.println( "\t" + vm.xpath("template/disk/source") );

    // Let's delete the VirtualMachine object.
    vm = null;

    // The reference is lost, but we can ask OpenNebula about the VM
```

(continues on next page)

(continued from previous page)

```
    // again. This time however, we are going to use the VM pool
    VirtualMachinePool vmPool = new VirtualMachinePool(oneClient);
    // Remember that we have to ask the pool to retrieve the information
    // from OpenNebula
    rc = vmPool.info();

    if(rc.isError())
        throw new Exception( rc.getErrorMessage() );

    System.out.println(
            "\nThese are all the Virtual Machines in the pool:");
    for ( VirtualMachine vmachine : vmPool )
    {
        System.out.println("\tID : " + vmachine.getId() +
                           ", Name : " + vmachine.getName() );

        // Check if we have found the VM we are looking for
        if ( vmachine.getId().equals( ""+newVMID ) )
        {
            vm = vmachine;
        }
    }

    // We have also some useful helpers for the actions you can perform
    // on a virtual machine, like suspend:
    rc = vm.suspend();
    System.out.println("\nTrying to suspend the VM " + vm.getId() +
                       " (should fail)...");

    // This is all the information you can get from the OneResponse:
    System.out.println("\tOpenNebula response");
    System.out.println("\t  Error:  " + rc.isError());
    System.out.println("\t  Msg:    " + rc.getMessage());
    System.out.println("\t  ErrMsg: " + rc.getErrorMessage());

    rc = vm.terminate();
    System.out.println("\nTrying to terminate the VM " +
                       vm.getId() + "...");

    System.out.println("\tOpenNebula response");
    System.out.println("\t  Error:  " + rc.isError());
    System.out.println("\t  Msg:    " + rc.getMessage());
    System.out.println("\t  ErrMsg: " + rc.getErrorMessage());


}
catch (Exception e)
{
    System.out.println(e.getMessage());
}
```

1.5.4 Compilation

To compile the Java OCA, untar the OpenNebula source, `cd` to the java directory and use the build script:

```
$ cd src/oca/java
$ ./build.sh -d
Compiling java files into class files...
Packaging class files in a jar...
Generating javadocs...
```

This command will compile and package the code in `jar/org.opennebula.client.jar`, and the javadoc will be created in `share/doc/`.

You might want to copy the .jar files to a more convenient directory. You could use /usr/lib/one/java/

```
$ sudo mkdir /usr/lib/one/java/
$ sudo cp jar/* lib/* /usr/lib/one/java/
```

1.6 OneFlow Specification

The OpenNebula OneFlow API is a RESTful service to create, control and monitor services composed of interconnected Virtual Machines with deployment dependencies between them. Each group of Virtual Machines is deployed and managed as a single entity, and is completely integrated with the advanced OpenNebula user and group management. There are two kind of resources; services templates and services. All data is sent and received as JSON.

This guide is intended for developers. The OpenNebula distribution includes a cli to interact with OneFlow and it is also fully integrated in the Sunstone GUI

1.6.1 Authentication & Authorization

User authentication will be HTTP Basic access authentication. The credentials passed should be the User name and password.

```
$ curl -u "username:password" https://oneflow.server
```

1.6.2 Return Codes

The OneFlow API uses the following subset of HTTP Status codes:

- **200 OK** : The request has succeeded.
- **201 Created** : Request was successful and a new resource has being created
- **202 Accepted** : The request has been accepted for processing, but the processing has not been completed
- **204 No Content** : The request has been accepted for processing, but no info in the response
- **400 Bad Request** : Malformed syntax
- **401 Unauthorized** : Bad authentication
- **403 Forbidden** : Bad authorization
- **404 Not Found** : Resource not found
- **500 Internal Server Error** : The server encountered an unexpected condition which prevented it from fulfilling the request.
- **501 Not Implemented** : The functionality requested is not supported

```
> POST /service_template HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: onflow.server:2474
>
< HTTP/1.1 400 Bad Request
< Content-Type: text/html;charset=utf-8
< Content-Type:application/json;charset=utf-8
< Content-Length: 40
<
{
  "error": {
    "message": "Role 'worker' 'cardinality' must be greater than or equal to 'min_vms'
↪"
  }
}
```

The methods specified below are described without taking into account **4xx** (can be inferred from authorization information in section above) and **5xx** errors (which are method independent). HTTP verbs not defined for a particular entity will return a **501 Not Implemented**.

1.6.3 Methods

Service

Method	URL	Meaning / Entity Body	Response
GET	`/service`	**List** the contents of the `SERVICE` collection.	**200 OK**: A JSON representation of the collection in the http body
GET	`/service/<id>`	**Show** the `SERVICE` resource identified by <id>	**200 OK**: A JSON representation of the collection in the http body
DELETE	`service/<id>`	**Delete** the `SERVICE` resource identified by <id>	**204**:
POST	`/service/<id>/action`	**Perform** an action on the `SERVICE` resource identified by <id>. Available actions: shutdown, recover, chown, chgrp, chmod	**201**:
PUT	`/service/<id>/role/<name>`	**Update** the `ROLE` identified by <name> of the `SERVICE` resource identified by <id>. Currently the only attribute that can be updated is the cardinality.	**200 OK**:
POST	`/service/<id>/role/<name>/action`	**Perform** an action on all the Virtual Machines belonging to the `ROLE` identified by <name> of the `SERVICE` resource identified by <id>. Available actions: shutdown, shutdown-hard, undeploy, undeploy-hard, hold, release, stop, suspend, resume, boot, delete, delete-recreate, reboot, reboot-hard, poweroff, poweroff-hard, snapshot-create	**201**:

Service Template

Method	URL	Meaning / Entity Body	Response
GET	`/service_template`	**List** the contents of the `SERVICE_TEMPLATE` collection.	**200 OK**: A JSON representation of the collection in the http body
GET	`/service_template/<id>`	**Show** the `SERVICE_TEMPLATE` resource identified by <id>	**200 OK**: A JSON representation of the collection in the http body
DELETE	`service_template/<id>`	**Delete** the `SERVICE_TEMPLATE` resource identified by <id>	**204**:
POST	`/service_template`	**Create** a new `SERVICE_TEMPLATE` resource.	**201 Created**: A JSON representation of the new `SERVICE_TEMPLATE` resource in the http body
PUT	`/service_template/<id>`	**Update** the `SERVICE_TEMPLATE` resource identified by <id>.	**200 OK**:
POST	`/service_template/<id>/action`	**Perform** an action on the `SERVICE_TEMPLATE` resource identified by <id>. Available actions: instantiate, chown, chgrp, chmod	**201**:

1.6.4 Resource Representation

Service Schema

A Service is defined with JSON syntax templates.

Attribute	Type	Mandatory	Description
name	string	No	Name of the Service
deployment	string	No	Deployment strategy: **none**: All roles are deployed at the same time **straight**: Each Role is deployed when all its parent Roles are running Defaults to none
shutdown_action	string	No	VM shutdown action: 'shutdown' or 'shutdown-hard'. If it is not set, the default set in oneflow-server.conf will be used
ready_status_gate	boolean	No	If ready_status_gate is set to true, a VM will only be considered to be in running state the following points are true: VM is in running state for OpenNebula. Which specifically means that LCM_STATE==3 and STATE>=3; The VM has READY=YES in the user template, this can be reported by the VM using OneGate.
roles	array of Roles	Yes	Array of Roles, see below

Each Role is defined as:

Attribute	Type	Mandatory	Description
name	string	Yes	Role name, only word characters (letter, number, underscore) are allowed
cardinality	integer	No	Number of VMs to deploy. Defaults to 1
vm_template	integer	Yes	OpenNebula VM Template ID. See the OpenNebula documentation for VM Templates
parents	array of string	No	Names of the roles that must be deployed before this one
shut-down_action	string	No	VM shutdown action: 'shutdown' or 'shutdown-hard'. If it is not set, the one set for the Service will be used
min_vms	integer	No (Yes for elasticity)	Minimum number of VMs for elasticity adjustments
max_vms	integer	No (Yes for elasticity)	Maximum number of VMs for elasticity adjustments
cooldown	integer	No	Cooldown period duration after a scale operation, in seconds. If it is not set, the default set in oneflow-server.conf will be used
elastic-ity_policies	array of Policies	No	Array of Elasticity Policies, see below
sched-uled_policies	array of Policies	No	Array of Scheduled Policies, see below

To define a elasticity policy:

At-tribute	Type	Manda-tory	Description
type	string	Yes	Type of adjustment. Values: CHANGE, CARDINALITY, PERCENTAGE_CHANGE
adjust	in-te-ger	Yes	Positive or negative adjustment. Its meaning depends on 'type'
min_adjust_step	in-te-ger	No	Optional parameter for PERCENTAGE_CHAGE adjustment type. If present, the policy will change the cardinality by at least the number of VMs set in this attribute.
expres-sion	string	Yes	Expression to trigger the elasticity
pe-riod_number	in-te-ger	No	Number of periods that the expression must be true before the elasticity is triggered
period	in-te-ger	No	Duration, in seconds, of each period in period_duration
cooldown	in-te-ger	No	Cooldown period duration after a scale operation, in seconds. If it is not set, the one set for the Role will be used

And each scheduled policy is defined as:

Attribute	Type	Mandatory	Description
type	string	Yes	Type of adjustment. Values: CHANGE, CARDINALITY, PERCENTAGE_CHANGE
adjust	integer	Yes	Positive or negative adjustment. Its meaning depends on 'type'
min_adjust_step	integer	No	Optional parameter for PERCENTAGE_CHAGE adjustment type. If present, the policy will change the cardinality by at least the number of VMs set in this attribute.
recurrence	string	No	Time for recurring adjustements. Time is specified with the Unix cron syntax
start_time	string	No	Exact time for the adjustement

```
{
  :type => :object,
  :properties => {
    'name' => {
      :type => :string,
      :required => true
    },
    'deployment' => {
      :type => :string,
      :enum => %w{none straight},
      :default => 'none'
    },
    'shutdown_action' => {
      :type => :string,
      :enum => %w{shutdown shutdown-hard},
      :required => false
    },
    'roles' => {
      :type => :array,
      :items => ROLE_SCHEMA,
      :required => true
    },
    'custom_attrs' => {
      :type => :object,
      :properties => {
      },
      :required => false
    },
    'ready_status_gate' => {
      :type => :boolean,
      :required => false
    }
  }
}
```

Role Schema

```
{
  :type => :object,
  :properties => {
```

(continues on next page)

(continued from previous page)

```
    'name' => {
      :type => :string,
      :required => true
    },
    'cardinality' => {
      :type => :integer,
      :default => 1,
      :minimum => 0
    },
    'vm_template' => {
      :type => :integer,
      :required => true
    },
    'vm_template_contents' => {
      :type => :string,
      :required => false
    },
    'parents' => {
      :type => :array,
      :items => {
        :type => :string
      }
    },
    'shutdown_action' => {
      :type => :string,
      :enum => ['shutdown', 'shutdown-hard']},
      :required => false
    },
    'min_vms' => {
      :type => :integer,
      :required => false,
      :minimum => 0
    },
    'max_vms' => {
      :type => :integer,
      :required => false,
      :minimum => 0
    },
    'cooldown' => {
      :type => :integer,
      :required => false,
      :minimum => 0
    },
    'elasticity_policies' => {
      :type => :array,
      :items => {
        :type => :object,
        :properties => {
          'type' => {
            :type => :string,
            :enum => ['CHANGE', 'CARDINALITY', 'PERCENTAGE_CHANGE'],
            :required => true
          },
          'adjust' => {
            :type => :integer,
            :required => true
          },
```

(continues on next page)

(continued from previous page)

```
          'min_adjust_step' => {
            :type => :integer,
            :required => false,
            :minimum => 1
          },
          'period_number' => {
            :type => :integer,
            :required => false,
            :minimum => 0
          },
          'period' => {
            :type => :integer,
            :required => false,
            :minimum => 0
          },
          'expression' => {
            :type => :string,
            :required => true
          },
          'cooldown' => {
            :type => :integer,
            :required => false,
            :minimum => 0
          }
        }
      }
    },
    'scheduled_policies' => {
      :type => :array,
      :items => {
        :type => :object,
        :properties => {
          'type' => {
            :type => :string,
            :enum => ['CHANGE', 'CARDINALITY', 'PERCENTAGE_CHANGE'],
            :required => true
          },
          'adjust' => {
            :type => :integer,
            :required => true
          },
          'min_adjust_step' => {
            :type => :integer,
            :required => false,
            :minimum => 1
          },
          'start_time' => {
            :type => :string,
            :required => false
          },
          'recurrence' => {
            :type => :string,
            :required => false
          }
        }
      }
    }
```

(continues on next page)

(continued from previous page)

```
  }
}
```

Action Schema

```
{
  :type => :object,
  :properties => {
    'action' => {
      :type => :object,
      :properties => {
        'perform' => {
          :type => :string,
          :required => true
        },
        'params' => {
          :type => :object,
            :required => false
          }
        }
      }
    }
  }
}
```

1.6.5 Examples

Create a New Service Template

Method	URL	Meaning / Entity Body	Response
POST	/service_template	**Create** a new SERVICE_TEMPLATE resource.	**201 Created**: A JSON representation of the new SERVICE_TEMPLATE resource in the http body

```
curl http://127.0.0.1:2474/service_template -u 'oneadmin:password' -v --data '{
  "name":"web-application",
  "deployment":"straight",
  "roles":[
    {
      "name":"frontend",
      "cardinality":"1",
      "vm_template":"0",
      "shutdown_action":"shutdown",
      "min_vms":"1",
      "max_vms":"4",
      "cooldown":"30",
      "elasticity_policies":[
        {
          "type":"PERCENTAGE_CHANGE",
          "adjust":"20",
          "min_adjust_step":"1",
```

(continues on next page)

(continued from previous page)

```
          "expression":"CUSTOM_ATT>40",
          "period":"3",
          "period_number":"30",
          "cooldown":"30"
        }
      ],
      "scheduled_policies":[
        {
          "type":"CHANGE",
          "adjust":"4",
          "recurrence":"0 2 1-10 * *"
        }
      ]
    },
    {
      "name":"worker",
      "cardinality":"2",
      "vm_template":"0",
      "shutdown_action":"shutdown",
      "parents":[
        "frontend"
      ],
      "min_vms":"2",
      "max_vms":"10",
      "cooldown":"240",
      "elasticity_policies":[
        {
          "type":"CHANGE",
          "adjust":"5",
          "expression":"ATT=3",
          "period":"5",
          "period_number":"60",
          "cooldown":"240"
        }
      ],
      "scheduled_policies":[
      ]
    }
  ],
  "shutdown_action":"shutdown"
}'
```

```
> POST /service_template HTTP/1.1
> Authorization: Basic b25lYWRtaW46b23lbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: oneflow.server:2474
> Accept: */*
> Content-Length: 771
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 201 Created
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 1990
< X-Frame-Options: sameorigin
```

(continues on next page)

(continued from previous page)

```
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT": {
    "TEMPLATE": {
      "BODY": {
        "deployment": "straight",
        "name": "web-application",
        "roles": [
          {
            "scheduled_policies": [
              {
                "adjust": 4,
                "type": "CHANGE",
                "recurrence": "0 2 1-10 * *"
              }
            ],
            "vm_template": 0,
            "name": "frontend",
            "min_vms": 1,
            "max_vms": 4,
            "cardinality": 1,
            "cooldown": 30,
            "shutdown_action": "shutdown",
            "elasticity_policies": [
              {
                "expression": "CUSTOM_ATT>40",
                "adjust": 20,
                "min_adjust_step": 1,
                "cooldown": 30,
                "period": 3,
                "period_number": 30,
                "type": "PERCENTAGE_CHANGE"
              }
            ]
          },
          {
            "scheduled_policies": [

            ],
            "vm_template": 0,
            "name": "worker",
            "min_vms": 2,
            "max_vms": 10,
            "cardinality": 2,
            "parents": [
              "frontend"
            ],
            "cooldown": 240,
            "shutdown_action": "shutdown",
            "elasticity_policies": [
              {
                "expression": "ATT=3",
                "adjust": 5,
                "cooldown": 240,
                "period": 5,
```

(continues on next page)

(continued from previous page)

```
                "period_number": 60,
                "type": "CHANGE"
              }
            ]
          }
        ],
        "shutdown_action": "shutdown"
      }
    },
    "TYPE": "101",
    "GNAME": "oneadmin",
    "NAME": "web-application",
    "GID": "0",
    "ID": "4",
    "UNAME": "oneadmin",
    "PERMISSIONS": {
      "OWNER_A": "0",
      "OWNER_M": "1",
      "OWNER_U": "1",
      "OTHER_A": "0",
      "OTHER_M": "0",
      "OTHER_U": "0",
      "GROUP_A": "0",
      "GROUP_M": "0",
      "GROUP_U": "0"
    },
    "UID": "0"
  }
```

Get Detailed Information of a Given Service Template

Method	URL	Meaning / Entity Body	Response
GET	/ service_templates/ <id>	**Show** the SERVICE_TEMPLATE resource identified by <id>	**200 OK**: A JSON representation of the collection in the http body

```
curl -u 'oneadmin:opennebula' http://127.0.0.1:2474/service_template/4 -v
```

```
> GET /service_template/4 HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 1990
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
```

(continues on next page)

(continued from previous page)

```
{
  "DOCUMENT": {
    "TEMPLATE": {
      "BODY": {
        "deployment": "straight",
        "name": "web-application",
        "roles": [
          {
            "scheduled_policies": [
              {
                "adjust": 4,
                "type": "CHANGE",
                "recurrence": "0 2 1-10 * *"
              }
            ],
            "vm_template": 0,
            ...
```

List the Available Service Templates

Method	URL	Meaning / Entity Body	Response
GET	/service_template	**List** the contents of the SERVICE_TEMPLATE collection.	**200 OK**: A JSON representation of the collection in the http body

```
curl -u 'oneadmin:opennebula' http://127.0.0.1:2474/service_template -v
```

```
> GET /service_template HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 6929
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT_POOL": {
    "DOCUMENT": [
      {
        "TEMPLATE": {
          "BODY": {
            "deployment": "straight",
            "name": "web-server",
            "roles": [
              {
                "scheduled_policies": [
                  {
```

(continues on next page)

(continued from previous page)

```
                    "adjust": 4,
                    "type": "CHANGE",
                    "recurrence": "0 2 1-10 * *"
                  }
                ],
                "vm_template": 0,
                "name": "frontend",
                "min_vms": 1,
                "max_vms": 4,
                "cardinality": 1,
                "cooldown": 30,
                "shutdown_action": "shutdown",
                "elasticity_policies": [
                  {
                ...
```

Update a Given Template

Method	URL	Meaning / Entity Body	Response
PUT	`/service_template/<id>`	**Update** the `SERVICE_TEMPLATE` resource identified by <id>.	**200 OK**:

```
curl http://127.0.0.1:2474/service_template/4 -u 'oneadmin:opennebula' -v -X PUT --
↪data '{
  "name":"web-application",
  "deployment":"straight",
  "roles":[
    {
      "name":"frontend",
      "cardinality":"1",
      "vm_template":"0",
      "shutdown_action":"shutdown-hard",
      "min_vms":"1",
      "max_vms":"4",
      "cooldown":"30",
      "elasticity_policies":[
        {
          "type":"PERCENTAGE_CHANGE",
          "adjust":"20",
          "min_adjust_step":"1",
          "expression":"CUSTOM_ATT>40",
          "period":"3",
          "period_number":"30",
          "cooldown":"30"
        }
      ],
      "scheduled_policies":[
        {
          "type":"CHANGE",
          "adjust":"4",
          "recurrence":"0 2 1-10 * *"
        }
```

(continues on next page)

(continued from previous page)

```
      ]
    },
    {
      "name":"worker",
      "cardinality":"2",
      "vm_template":"0",
      "shutdown_action":"shutdown",
      "parents":[
        "frontend"
      ],
      "min_vms":"2",
      "max_vms":"10",
      "cooldown":"240",
      "elasticity_policies":[
        {
          "type":"CHANGE",
          "adjust":"5",
          "expression":"ATT=3",
          "period":"5",
          "period_number":"60",
          "cooldown":"240"
        }
      ],
      "scheduled_policies":[
      ]
    }
  ],
  "shutdown_action":"shutdown"
}'
```

```
> PUT /service_template/4 HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
> Content-Length: 1219
> Content-Type: application/x-www-form-urlencoded
> Expect: 100-continue
>
* Done waiting for 100-continue
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 1995
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT": {
    "TEMPLATE": {
      "BODY": {
        "deployment": "straight",
        "name": "web-application",
        "roles": [
```

(continues on next page)

(continued from previous page)

```
          {
            "scheduled_policies": [
              {
                "adjust": 4,
                "type": "CHANGE",
                "recurrence": "0 2 1-10 * *"
              }
            ],
            "vm_template": 0,
            "name": "frontend",
            "min_vms": 1,
            "max_vms": 4,
            "cardinality": 1,
            "cooldown": 30,
            "shutdown_action": "shutdown-hard",
            ...
```

Instantiate a Given Template

Method	URL	Meaning / Entity Body	Response
POST	/service_template/<id>/action	**Perform** an action on the SERVICE_TEMPLATE resource identified by <id>. Available actions: instantiate, chown, chgrp, chmod	**201**:

Available actions:

- instantiate
- chown
- chmod
- chgrp

```
curl http://127.0.0.1:2474/service_template/4/action -u 'oneadmin:opennebula' -v -X␣
↪POST --data '{
  "action": {
    "perform":"instantiate"
  }
}'
```

```
> POST /service_template/4/action HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
> Content-Length: 49
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 201 Created
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
```

(continues on next page)

(continued from previous page)

```
< Content-Length: 2015
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT": {
    "TEMPLATE": {
      "BODY": {
        "deployment": "straight",
        "name": "web-application",
        "roles": [
          {
            "scheduled_policies": [
              {
                "adjust": 4,
                "type": "CHANGE",
                "recurrence": "0 2 1-10 * *"
              }
            ],
            "vm_template": 0,
```

Delete a Given Template

Method	URL	Meaning / Entity Body	Response
DELETE	/service_template/<id>	**Delete** the SERVICE_TEMPLATE resource identified by <id>	**204**:

```
curl http://127.0.0.1:2474/service_template/4 -u 'oneadmin:opennebula' -v -X DELETE
```

```
> DELETE /service_template/3 HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
>
< HTTP/1.1 204 No Content
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 0
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
```

Get Detailed Information of a Given Service

Method	URL	Meaning / Entity Body	Response
GET	/service/<id>	**Show** the SERVICE resource identified by <id>	**200 OK**: A JSON representation of the collection in the http body

```
curl http://127.0.0.1:2474/service/5 -u 'oneadmin:opennebula' -v
```

```
> GET /service/5 HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 11092
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT": {
    "TEMPLATE": {
      "BODY": {
        "deployment": "straight",
        "name": "web-application",
        "roles": [
          {
            "scheduled_policies": [
              {
                "adjust": 4,
                "last_eval": 1374676803,
                "type": "CHANGE",
                "recurrence": "0 2 1-10 * *"
              }
            ],
            "vm_template": 0,
            "disposed_nodes": [

            ],
            "name": "frontend",
            "min_vms": 1,
            "nodes": [
              {
                "deploy_id": 12,
                "vm_info": {
                  "VM": {
                    "CPU": "33",
                    "TEMPLATE": {
                      "CPU": "1",
                      "CONTEXT": {
                        "TARGET": "hda",
                        "NETWORK": "YES",
                        "DISK_ID": "0"
                      },
                      "MEMORY": "1024",
                      "TEMPLATE_ID": "0",
                      "VMID": "12"
                    },
                    "GNAME": "oneadmin",
```

(continues on next page)

(continued from previous page)


```
                    "RESCHED": "0",
                    "NET_RX": "1300",
                    "NAME": "frontend_0_(service_5)",
                    "ETIME": "0",
                    "USER_TEMPLATE": {
                      "SERVICE_ID": "5",
                      "ROLE_NAME": "frontend"
                    },
                    "GID": "0",
                    "LAST_POLL": "1374676793",
                    "MEMORY": "786432",
                    "HISTORY_RECORDS": {
                      "HISTORY": {
                        "RETIME": "0",
                        "TMMAD": "dummy",
                        "DS_LOCATION": "/var/tmp/one_install/var//datastores",
                        "SEQ": "0",
                        "VNMMAD": "dummy",
                        "ETIME": "0",
                        "PETIME": "1374676347",
                        "HOSTNAME": "vmx_dummy",
                        "VMMMAD": "dummy",
                        "ESTIME": "0",
                        "HID": "2",
                        "EETIME": "0",
                        "OID": "12",
                        "STIME": "1374676347",
                        "DS_ID": "0",
                        "ACTION": "0",
                        "RSTIME": "1374676347",
                        "REASON": "0",
                        "PSTIME": "1374676347"
                      }
                    },
                    "ID": "12",
                    "DEPLOY_ID": "vmx_dummy:frontend_0_(service_5):dummy",
                    "NET_TX": "800",
                    "UNAME": "oneadmin",
                    "LCM_STATE": "3",
                    "STIME": "1374676345",
                    "UID": "0",
                    "PERMISSIONS": {
                      "OWNER_U": "1",
                      "OWNER_M": "1",
                      "OWNER_A": "0",
                      "GROUP_U": "0",
                      "GROUP_M": "0",
                      "GROUP_A": "0",
                      "OTHER_U": "0",
                      "OTHER_M": "0",
                      "OTHER_A": "0"
                    },
                    "STATE": "3"
                  }
                }
              }
            ],
```


(continues on next page)

(continued from previous page)


```
            "last_vmname": 1,
            "max_vms": 4,
            "cardinality": 1,
            "cooldown": 30,
            "shutdown_action": "shutdown-hard",
            "state": "2",
            "elasticity_policies": [
              {
                "expression": "CUSTOM_ATT>40",
                "true_evals": 0,
                "adjust": 20,
                "min_adjust_step": 1,
                "last_eval": 1374676803,
                "cooldown": 30,
                "expression_evaluated": "CUSTOM_ATT[--] > 40",
                "period": 3,
                "period_number": 30,
                "type": "PERCENTAGE_CHANGE"
              }
            ]
          },
          {
            "scheduled_policies": [

            ],
            "vm_template": 0,
            "disposed_nodes": [

            ],
            "name": "worker",
            "min_vms": 2,
            "nodes": [
              {
                "deploy_id": 13,
                "vm_info": {
                  "VM": {
                    "CPU": "9",
                    "TEMPLATE": {
                      "CPU": "1",
                      "CONTEXT": {
                        "TARGET": "hda",
                        "NETWORK": "YES",
                        "DISK_ID": "0"
                      },
                      "MEMORY": "1024",
                      "TEMPLATE_ID": "0",
                      "VMID": "13"
                    },
                    "GNAME": "oneadmin",
                    "RESCHED": "0",
                    "NET_RX": "1600",
                    "NAME": "worker_0_(service_5)",
                    "ETIME": "0",
                    "USER_TEMPLATE": {
                      "SERVICE_ID": "5",
                      "ROLE_NAME": "worker"
                    },
```


(continues on next page)

(continued from previous page)


```
                    "GID": "0",
                    "LAST_POLL": "1374676783",
                    "MEMORY": "545259",
                    "HISTORY_RECORDS": {
                      "HISTORY": {
                        "RETIME": "0",
                        "TMMAD": "dummy",
                        "DS_LOCATION": "/var/tmp/one_install/var//datastores",
                        "SEQ": "0",
                        "VNMMAD": "dummy",
                        "ETIME": "0",
                        "PETIME": "1374676377",
                        "HOSTNAME": "xen_dummy",
                        "VMMMAD": "dummy",
                        "ESTIME": "0",
                        "HID": "1",
                        "EETIME": "0",
                        "OID": "13",
                        "STIME": "1374676377",
                        "DS_ID": "0",
                        "ACTION": "0",
                        "RSTIME": "1374676377",
                        "REASON": "0",
                        "PSTIME": "1374676377"
                      }
                    },
                    "ID": "13",
                    "DEPLOY_ID": "xen_dummy:worker_0_(service_5):dummy",
                    "NET_TX": "600",
                    "UNAME": "oneadmin",
                    "LCM_STATE": "3",
                    "STIME": "1374676375",
                    "UID": "0",
                    "PERMISSIONS": {
                      "OWNER_U": "1",
                      "OWNER_M": "1",
                      "OWNER_A": "0",
                      "GROUP_U": "0",
                      "GROUP_M": "0",
                      "GROUP_A": "0",
                      "OTHER_U": "0",
                      "OTHER_M": "0",
                      "OTHER_A": "0"
                    },
                    "STATE": "3"
                  }
                }
              },
              {
                "deploy_id": 14,
                "vm_info": {
                  "VM": {
                    "CPU": "75",
                    "TEMPLATE": {
                      "CPU": "1",
                      "CONTEXT": {
                        "TARGET": "hda",
```


(continues on next page)

(continued from previous page)


```
                    "NETWORK": "YES",
                    "DISK_ID": "0"
                  },
                  "MEMORY": "1024",
                  "TEMPLATE_ID": "0",
                  "VMID": "14"
                },
                "GNAME": "oneadmin",
                "RESCHED": "0",
                "NET_RX": "1100",
                "NAME": "worker_1_(service_5)",
                "ETIME": "0",
                "USER_TEMPLATE": {
                  "SERVICE_ID": "5",
                  "ROLE_NAME": "worker"
                },
                "GID": "0",
                "LAST_POLL": "1374676783",
                "MEMORY": "471859",
                "HISTORY_RECORDS": {
                  "HISTORY": {
                    "RETIME": "0",
                    "TMMAD": "dummy",
                    "DS_LOCATION": "/var/tmp/one_install/var//datastores",
                    "SEQ": "0",
                    "VNMMAD": "dummy",
                    "ETIME": "0",
                    "PETIME": "1374676378",
                    "HOSTNAME": "kvm_dummy",
                    "VMMMAD": "dummy",
                    "ESTIME": "0",
                    "HID": "0",
                    "EETIME": "0",
                    "OID": "14",
                    "STIME": "1374676378",
                    "DS_ID": "0",
                    "ACTION": "0",
                    "RSTIME": "1374676378",
                    "REASON": "0",
                    "PSTIME": "1374676378"
                  }
                },
                "ID": "14",
                "DEPLOY_ID": "kvm_dummy:worker_1_(service_5):dummy",
                "NET_TX": "550",
                "UNAME": "oneadmin",
                "LCM_STATE": "3",
                "STIME": "1374676375",
                "UID": "0",
                "PERMISSIONS": {
                  "OWNER_U": "1",
                  "OWNER_M": "1",
                  "OWNER_A": "0",
                  "GROUP_U": "0",
                  "GROUP_M": "0",
                  "GROUP_A": "0",
                  "OTHER_U": "0",
```


(continues on next page)

(continued from previous page)


```
                    "OTHER_M": "0",
                    "OTHER_A": "0"
                  },
                  "STATE": "3"
                }
              }
            }
          ],
          "last_vmname": 2,
          "max_vms": 10,
          "cardinality": 2,
          "parents": [
            "frontend"
          ],
          "cooldown": 240,
          "shutdown_action": "shutdown",
          "state": "2",
          "elasticity_policies": [
            {
              "expression": "ATT=3",
              "true_evals": 0,
              "adjust": 5,
              "last_eval": 1374676803,
              "cooldown": 240,
              "expression_evaluated": "ATT[--] = 3",
              "period": 5,
              "period_number": 60,
              "type": "CHANGE"
            }
          ]
        }
      ],
      "log": [
        {
          "message": "New state: DEPLOYING",
          "severity": "I",
          "timestamp": 1374676345
        },
        {
          "message": "New state: RUNNING",
          "severity": "I",
          "timestamp": 1374676406
        }
      ],
      "shutdown_action": "shutdown",
      "state": 2
    }
  },
  "TYPE": "100",
  "GNAME": "oneadmin",
  "NAME": "web-application",
  "GID": "0",
  "ID": "5",
  "UNAME": "oneadmin",
  "PERMISSIONS": {
    "OWNER_A": "0",
    "OWNER_M": "1",
```


(continues on next page)

(continued from previous page)

```
      "OWNER_U": "1",
      "OTHER_A": "0",
      "OTHER_M": "0",
      "OTHER_U": "0",
      "GROUP_A": "0",
      "GROUP_M": "0",
      "GROUP_U": "0"
    },
    "UID": "0"
  }
```

List the Available Services

Method	URL	Meaning / Entity Body	Response
GET	`/service`	**List** the contents of the `SERVICE` collection.	**200 OK**: A JSON representation of the collection in the http body

```
curl http://127.0.0.1:2474/service -u 'oneadmin:opennebula' -v
```

```
> GET /service HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 12456
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
<
{
  "DOCUMENT_POOL": {
    "DOCUMENT": [
      {
        "TEMPLATE": {
          "BODY": {
            "deployment": "straight",
            "name": "web-application",
            "roles": [
              {
                "scheduled_policies": [
                  {
                    "adjust": 4,
                    "last_eval": 1374676986,
                    "type": "CHANGE",
                    "recurrence": "0 2 1-10 * *"
                  }
                ],
                ...
```

Perform an Action on a Given Service

Method	URL	Meaning / Entity Body	Response
POST	`/service/<id>/action`	**Perform** an action on the `SERVICE` resource identified by <id>.	**201**:

Available actions:

- **shutdown: Shutdown a service.**
  - From RUNNING or WARNING shuts down the Service
- **recover: Recover a failed service, cleaning the failed VMs.**
  - From FAILED_DEPLOYING continues deploying the Service
  - From FAILED_SCALING continues scaling the Service
  - From FAILED_UNDEPLOYING continues shutting down the Service
  - From COOLDOWN the Service is set to running ignoring the cooldown duration
  - From WARNING failed VMs are deleted, and new VMs are instantiated
- chown
- chmod
- chgrp

```
curl http://127.0.0.1:2474/service/5/action -u 'oneadmin:opennebula' -v -X POST --
↪data '{
  "action": {
    "perform":"shutdown"
  }
}'
```

```
curl http://127.0.0.1:2474/service/5/action -u 'oneadmin:opennebula' -v -X POST --
↪data '{
  "action": {
    "perform":"chgrp",
    "params" : {
      "group_id" : 2
    }
  }
}'
```

Update the Cardinality of a Given Role

Method	URL	Meaning / Entity Body	Response
PUT	`/service/<id>/role/<name>`	**Update** the `ROLE` identified by <name> of the `SERVICE` resource identified by <id>. Currently the only attribute that can be updated is the cardinality.	**200 OK**:

You can force a cardinality outside the defined range with the force param.

```
curl http://127.0.0.1:2474/service/5/role/frontend -u 'oneadmin:opennebula' -X PUT -v␣
↪--data '{
  "cardinality" : 2,
  "force" : true
}'
```

```
> PUT /service/5/role/frontend HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
> Content-Length: 41
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 0
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
```

Perform an Action on All the VMs of a Given Role

Method	URL	Meaning / Entity Body	Response
POST	`/service/<id>/role/<name>/action`	**Perform** an action on all the Virtual Machines belonging to the `ROLE` identified by <name> of the `SERVICE` resource identified by <id>.	**201**:

You can use this call to perform a VM action on all the Virtual Machines belonging to a role. For example, if you want to suspend the Virtual Machines of the worker Role:

These are the commands that can be performed:

- `shutdown`
- `shutdown-hard`
- `undeploy`
- `undeploy-hard`
- `hold`
- `release`
- `stop`
- `suspend`
- `resume`
- `boot`
- `delete`
- `delete-recreate`

- `reboot`
- `reboot-hard`
- `poweroff`
- `poweroff-hard`
- `snapshot-create`

Instead of performing the action immediately on all the VMs, you can perform it on small groups of VMs with these options:

- `period`: Seconds between each group of actions
- `number`: Number of VMs to apply the action to each period

```
curl http://127.0.0.1:2474/service/5/role/frontend/action -u 'oneadmin:opennebula' -v␣
↪-X POST --data '{
  "action": {
    "perform":"stop",
    "params" : {
      "period" : 60,
      "number" : 2
    }
  }
}'
```

```
> POST /service/5/role/frontend/action HTTP/1.1
> Authorization: Basic b25lYWRtaW46b3Blbm5lYnVsYQ==
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0 zlib/
↪1.2.3 libidn/1.18 libssh2/1.4.2
> Host: 127.0.0.1:2474
> Accept: */*
> Content-Length: 106
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 201 Created
< Content-Type: text/html;charset=utf-8
< X-XSS-Protection: 1; mode=block
< Content-Length: 57
< X-Frame-Options: sameorigin
< Connection: keep-alive
< Server: thin 1.2.8 codename Black Keys
```

1.7 Go OpenNebula Cloud API

This page contains the OpenNebula Cloud API Specification for Go. It has been designed as a wrapper for the *XML-RPC methods*, with some basic helpers. This means that you should be familiar with the XML-RPC API and the XML formats returned by the OpenNebula core. As stated in the *XML-RPC documentation*, you can download the *XML Schemas (XSD) here*.

Go OpenNebula Cloud API cover the resources lists below:

Resource	URL
ACL	acl.go
Cluster	cluster.go
Datastore	datastore.go
Document	document.go
Group	group.go
Host	host.go
Image	image.go
Template	template.go
User	user.go
VDC	vdc.go
Vnet	virtualnetwork.go
VMs	vm.go
Zone	zone.go

1.7.1 Download

The source code can be downloaded from the OpenNebula repository.

1.7.2 Usage

To use the OpenNebula Cloud API for Go in your Go project, you have to import goca at your project as the example below and make a `go get`.

1.7.3 Code Sample

The example below show how get the information of a running VM, print its name, and power it off. It then builds a new OpenNebula template and prints its string representation.

```
package main

import (
    "fmt"
    "github.com/OpenNebula/one/src/oca/go/src/goca"
    "log"
    "os"
    "strconv"
)

func main() {
    id, _ := strconv.Atoi(os.Args[1])

    vm := goca.NewVM(uint(id))

    err := vm.Info()
    if err != nil {
        log.Fatal(err)
    }

    name, _ := vm.XPath("/VM/NAME")
    if err != nil {
```

(continues on next page)

(continued from previous page)

```
        log.Fatal(err)
    }

    fmt.Println(name)

    // Poweroff the VM
    err = vm.PoweroffHard()
    if err != nil {
        log.Fatal(err)
    }

    // Create a new Template
    template := goca.NewTemplateBuilder()

    template.AddValue("cpu", 1)
    template.AddValue("memory", "64")
    vector := template.NewVector("disk")
    vector.AddValue("image_id", "119")
    vector.AddValue("dev_prefix", "vd")
    vector = template.NewVector("nic")
    vector.AddValue("network_id", "3")
    vector.AddValue("model", "virtio")
    template.AddValue("vcpu", "2")

    fmt.Println(template)
}
```

Limitations

Go OpenNebula Cloud API doesn't cover the resources list below:

Resource	URL
Marketplace	http://docs.opennebula.org/5.9/integration/system_interfaces/api.html#onemarket
Marketapp	http://docs.opennebula.org/5.9/integration/system_interfaces/api.html#onemarketapp
Security Groups	http://docs.opennebula.org/5.9/integration/system_interfaces/api.html#onesecgroup
VM Groups	http://docs.opennebula.org/5.9/integration/system_interfaces/api.html#onevmgroup
Virtual Router	http://docs.opennebula.org/5.9/integration/system_interfaces/api.html#onevrouter

CHAPTER

TWO

INFRASTRUCTURE INTEGRATION

2.1 Overview

The interactions between OpenNebula and the Cloud infrastructure are performed by specific drivers. Each one addresses a particular area:

- **Storage**. The OpenNebula core issue abstracts storage operations (e.g. clone or delete) that are implemented by specific programs that can be replaced or modified to interface special storage backends and file-systems.
- **Virtualization**. The interaction with the hypervisors are also implemented with custom programs to boot, stop or migrate a virtual machine. This allows you to specialize each VM operation so to perform custom operations.
- **Monitoring**. Monitoring information is also gathered by external probes. You can add additional probes to include custom monitoring metrics that can later be used to allocate virtual machines or for accounting purposes.
- **Authorization**. OpenNebula can be also configured to use an external program to authorize and authenticate user requests. In this way, you can implement any access policy to Cloud resources.
- **Networking**. The hypervisor is also prepared with the network configuration for each Virtual Machine.

Use the driver interfaces if you need OpenNebula to interface any specific storage, virtualization, monitoring or authorization system already deployed in your datacenter or to tune the behavior of the standard OpenNebula drivers.

2.1.1 How Should I Read This Chapter

You should be reading this Chapter if you are trying to extend OpenNebula functionality.

You can proceed to any of the following sections depending on which component you want to understand and extend the *virtualization system* (with a separate Section for *cloud bursting*), the *storage system*, the *information system*, the authentication system, the *network system* or the *marketplace drivers*. Also you might be interested in the *Hook mechanism*, a powerful way of integrating OpenNebua within your datacenter processes.

After this Chapter, congratulations! You finished OpenNebula.

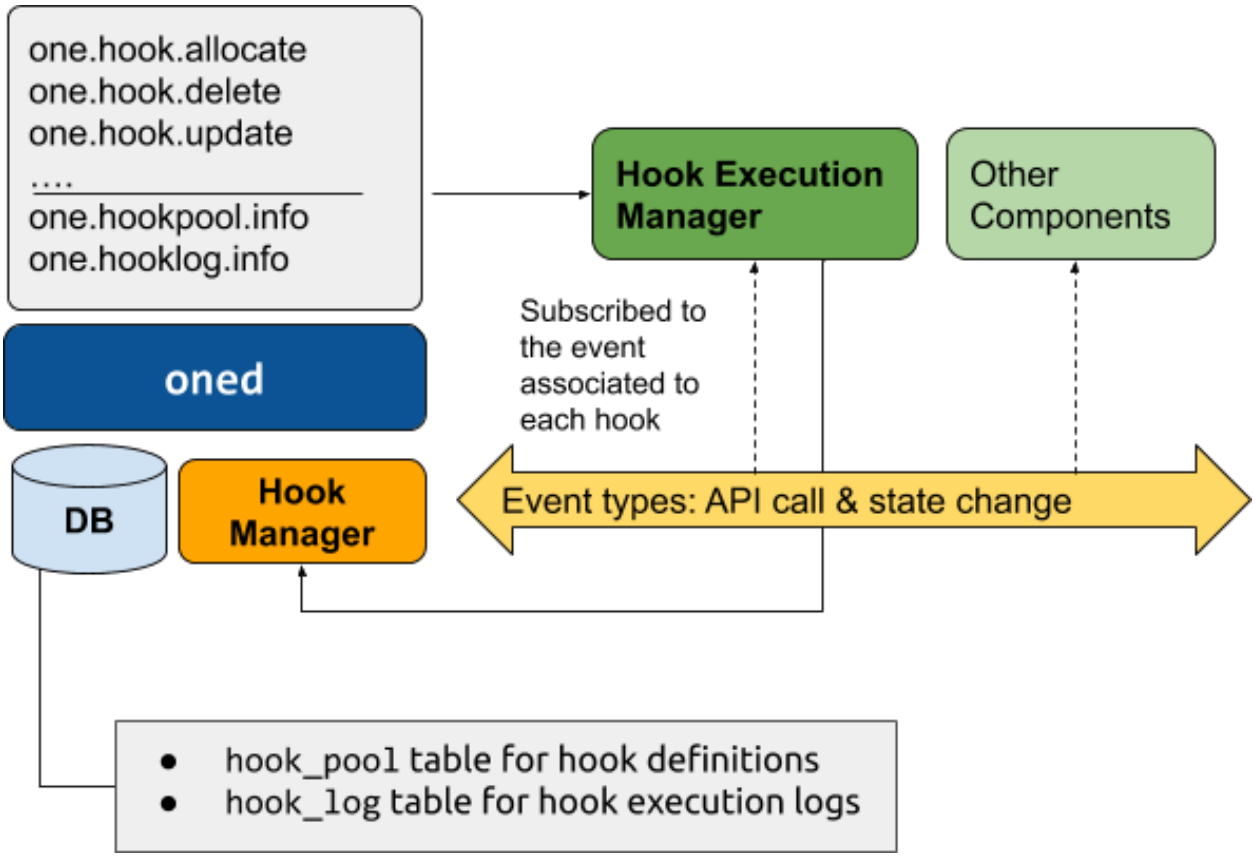
2.2 Using Hooks

The Hook subsystem enables the execution of custom scripts tied to a change in state in a particular resource, or API call. This opens a wide area of automation for system administrators to tailor their cloud infrastructures. It also features a logging mechanism that allows a convenient way to query the execution history or to retry the execution of a given hook.

2.2.1 Overview

The hook subsystem has two main components:

- **Hook Manager Driver**: it receives information about every event triggered by oned and publishes it to an event queue. Custom components can also use this queue to subscribe to oned events, *more information here*.
- **Hook Execution Manager** (HEM): It registers itself to the events that triggers the hooks defined in the system. When an event is received it takes care of executing the corresponding hook command.



Both components are started together with the OpenNebula daemon. Note that, provided the network communication is secure, you can grant network access to the event queue and hence deploy HEM in a different server.

2.2.2 Configuration

Hook Manager

Hook Manager configuration is set in `HM_MAD` section in `/etc/one/oned.conf`. The configuration attributes are described below:

Parameter	Description
Executable	Path of the hook driver executable, can be an absolute path or relative to $ONE_LOCATION/lib/mads (or /usr/lib/one/mads/ if OpenNebula was installed in /)
Arguments	Arguments for the driver executable, the following values are supported: • `--publisher-port`, `-p`: The port where the Hook Manager will publish the events reported by oned. • `--logger-port`, `-l`: The port where the Hook Manager will receive information about hook executions. • `--hwm`, `-h`: The HWM value for the publisher socket, more information can be found here. • `--bind`, `-b`: Address to bind the publisher socket.

Hook Execution Manager

Hook Execution Manager configuration is set in `/etc/one/onehem-server.conf`:

Parameter	Description
debug_level	Set the log debug level shown in `/var/log/one/onehem-server.log`
hook_base_path	Base location to look for hook scripts when commands use a relative path (default value `/var/lib/one/remotes/hooks`)
subscriber_endpoint	To subscribe for OpenNebula events, must match those in `HM_MAD` section of `oned.conf`.
replier_endpoint	To send hook execution results (reply to events) to oned, it must match those in `HM_MAD` section of `oned.conf`.
concurrency	Number of hooks executed simultaneously.

Hook Log

You can check the execution results of a hook using the hook log. The hook log stores a record for each execution including the standard output and error of the command, as well as the arguments used. This information is used to retry the execution of a given record.

The number of execution log records stored in the database for each hook can be configured in *oned.conf*. For example, to keep only the last 10 execution records of each hook use `LOG_RETENTION = 10`. This value can be tuned taking into account the number of hooks and how many times hooks are executed:

```
HOOK_LOG_CONF = [
    LOG_RETENTION = 10 ]
```

2.2.3 Hook Types

There are two types of hooks, **API hooks** (triggered on API calls) and **State hooks** (triggered on state change). The main attributes for both types are described in the table below:

Option	Mandatory	Description
NAME	YES	The name of the hook.
COMMAND	YES	The command that will be executed when hook is triggered. Typically a path to a script.
ARGUMENTS	NO	The arguments that will be passed to the script when the hook is triggered.
ARGUMENTS_STDIN	NO	If `yes` the ARGUMENTS will be passed through STDIN instead of as command line arguments.
TYPE	YES	The hook type either `api` or `state`.
TIMEOUT	NO	Hook execution timeout. If not used the timeout is infinite.

API Hooks

The API hooks are triggered after the execution of an API call. The specific attributes for API hooks are listed below:

Option	Mandatory	Description
CALL	YES	The API call which trigger the hook. For more info about API calls please check *XML-RPC API section.*

For API hooks, the `$API` keyword can be used in the `ARGUMENTS` attribute to get the information about the API call. If `$API` is used, a base64-encoded XML document containing the API call arguments and result will be passed to the hook command. The schema of the XML is defined here.

Note: If the API method defined in `CALL` is an `allocate` or `delete`, the `$API` document will also include the template of the corresponding resource.

The following example defines an API hook that executes the command `/var/lib/one/remotes/hooks/log_new_user.rb` whenever a new user is created:

```
NAME      = hook-API
TYPE      = api
COMMAND   = "log_new_user.rb"
ARGUMENTS = $API
CALL      = "one.user.allocate"
ARGUMENTS_STDIN = yes
```

State Hooks

The state hooks are only available for **Hosts** and **Virtual Machines** and they are triggered on specific state transitions. The specific attributes to define state hooks are:

Option	Mandatory	Description
RESOURCE	YES	Type of the resource, supported values are `HOST` and `VM`.
REMOTE	NO	If `yes` the hook will be executed in the host that triggered the hook (for Host hooks) or in the host where the VM is running (for VM hooks)
STATE	YES	The state that triggers the hook.
LCM_STATE	YES	The LCM state that triggers the hook (Only for VM hooks)
ON	YES	For `RESOURCE=VM`, shortcut to define common `STATE`/`LCM_STATE` pairs. Supported values are: CREATE, RUNNING, SHUTDOWN, STOP, DONE, UNKNOWN, CUSTOM

Warning: Note that `ON` is mandatory for VM hooks, use `ON=CUSTOM` with `STATE` and `LCM_STATE` to define hooks on specific state transitions.

For state hooks, `$TEMPLATE` can be used in the `ARGUMENTS` attribute to get the template (in XML format, base64 encoded) of the resource which triggered the hook. The XSD schema files for the XML document of each object are available here

The following examples define two state hooks for VMs and hosts:

```
# VM
NAME = hook-vm
TYPE = state
COMMAND = new_vm.rb
ARGUMENTS = $TEMPLATE
ON = PROLOG
RESOURCE = VM

# HOST
NAME = hook-host
TYPE = state
COMMAND = host-disabled.rb
STATE = DISABLED
RESOURCE = HOST
REMOTE = yes
```

Note: More info about VM and Host states can be found here and here

2.2.4 Managing Hooks

Hooks can be managed via the CLI through the `onehook` command and via the API. This section describes the common operations to control the life-cycle of a hook using the CLI.

Creating Hooks

In order to create a new hook you need to create a hook template:

```
$ cat > hook.tmpl << EOF
     NAME      = hook-vm
```

(continues on next page)

(continued from previous page)

```
    TYPE      = state
    COMMAND   = vm-pending.rb
    ARGUMENTS = "\$TEMPLATE pending"
    ON        = CUSTOM
    RESOURCE  = VM
    STATE     = PENDING
    LCM_STATE = LCM_INIT
EOF
```

Then, simply create the hook by running the following command:

```
$ onehook create hook.tmpl
  ID: 0
```

We have just created a hook which will be triggered each time a VM switch to PENDING state.

Checking Hook Execution

We can check the execution records of a hook by accessing its detailed information. For example, to get the execution history of the previous hook use `onehook show 0`:

```
$ onevm create --cpu 1 --memory 2 --name test
  ID: 0
$ onehook show 0
  HOOK 0 INFORMATION
  ID                : 0
  NAME              : hook-vm
  TYPE              : state
  LOCK              : None

  HOOK TEMPLATE
  ARGUMENTS="$TEMPLATE pending"
  COMMAND="vm-pending.rb"
  LCM_STATE="LCM_INIT"
  REMOTE="NO"
  RESOURCE="VM"
  STATE="PENDING"

  EXECUTION LOG
    ID    TIMESTAMP    EXECUTION
    0     09/23 15:10  ERROR (255)
```

We can see that there is an execution which have failed with error code 255. To get more information about a specific execution use the `-e` option:

```
$ onehook show 0 -e 0
  HOOK 0 INFORMATION
  ID                : 0
  NAME              : hook-vm
  TYPE              : state
  LOCK              : None

  HOOK EXECUTION RECORD
  EXECUTION ID      : 0
  TIMESTAMP         : 09/23 15:10:38
```

(continues on next page)

(continued from previous page)

```
  COMMAND           : /var/lib/one/remotes/hooks/vm-pending.rb PFZNPgogIDxJR...
↪8+CjwvVk0+ pending
  ARGUMENTS         :
  <VM>
  <ID>0</ID>
  <UID>0</UID>
  <GID>0</GID>
  <UNAME>oneadmin</UNAME>
  <GNAME>oneadmin</GNAME>
  <NAME>test</NAME>
  <PERMISSIONS>
      <OWNER_U>1</OWNER_U>
      <OWNER_M>1</OWNER_M>
      <OWNER_A>0</OWNER_A>
      <GROUP_U>0</GROUP_U>
      <GROUP_M>0</GROUP_M>
      <GROUP_A>0</GROUP_A>
      <OTHER_U>0</OTHER_U>
      <OTHER_M>0</OTHER_M>
      <OTHER_A>0</OTHER_A>
  </PERMISSIONS>
  <LAST_POLL>0</LAST_POLL>
  <STATE>1</STATE>
  <LCM_STATE>0</LCM_STATE>
  <PREV_STATE>1</PREV_STATE>
  <PREV_LCM_STATE>0</PREV_LCM_STATE>
  <RESCHED>0</RESCHED>
  <STIME>1569244238</STIME>
  <ETIME>0</ETIME>
  <DEPLOY_ID/>
  <MONITORING/>
  <TEMPLATE>
      <AUTOMATIC_REQUIREMENTS><![CDATA[!(PUBLIC_CLOUD = YES) & !(PIN_POLICY =␣
↪PINNED)]]></AUTOMATIC_REQUIREMENTS>
      <CPU><![CDATA[1]]></CPU>
      <MEMORY><![CDATA[2]]></MEMORY>
      <VMID><![CDATA[0]]></VMID>
  </TEMPLATE>
  <USER_TEMPLATE/>
  <HISTORY_RECORDS/>
  </VM> pending
  EXIT CODE         : 255

  EXECUTION STDOUT


  EXECUTION STDERR
  ERROR MESSAGE --8<------
  Internal error No such file or directory - /var/lib/one/remotes/hooks/vm-pending.rb
  ERROR MESSAGE ------>8--
```

The `EXECUTION STDERR` message shows the reason for the hook execution failure, the script does not exists:

```
Internal error No such file or directory - /var/lib/one/remotes/hooks/vm-pending.rb
```

Important: The hook log can be queried and filtered by several criteria using `onehook log`. More info about

`onehook log` command can be found running `onehook log --help`.

Retrying Hook Executions

We are going to fix the previous error, let's first create the `vm-pending.rb` script, and then retry the hook execution.

Note: Note that any hook execution can be retried regardless of it result.

```
$ vim /var/lib/one/remotes/hooks/vm-pending.rb
  #!/usr/bin/ruby
  puts "Executed!"

$ chmod 760 /var/lib/one/remotes/hooks/vm-pending.rb
$ onehook retry 0 0
$ onehook show 0
  HOOK 0 INFORMATION
  ID                : 0
  NAME              : hook-vm
  TYPE              : state
  LOCK              : None

  HOOK TEMPLATE
  ARGUMENTS="$TEMPLATE pending"
  COMMAND="vm-pending.rb"
  LCM_STATE="LCM_INIT"
  REMOTE="NO"
  RESOURCE="VM"
  STATE="PENDING"

  EXECUTION LOG
  ID       TIMESTAMP    RC    EXECUTION
  0        09/23 15:10  255   ERROR
  1        09/23 15:59    0   SUCCESS
```

Note the last successful execution record!

Important: When a hook execution is retried the same execution context is used, i.e. arguments and $TEMPLATE/$API values.

2.2.5 Developing Hooks

First thing you need to decide is the type of hook you are interested in, being it API or STATE hooks. Each type of hook is triggered by a different event and requires/provides different runtime information.

In this section you'll find two simple script hooks (in ruby) for each type. This examples are good starting points for developing custom hooks.

API Hook example

This script prints to stdout the result of one.user.create API call and the username of new user.

```
# Hook template
#
# NAME = user-create
# TYPE = api
# COMMAND = "user_create_hook.rb"
# ARGUMENTS = "$API"
# CALL = "one.user.allocate"

#!/usr/bin/ruby

require 'base64'
require 'nokogiri'

#api_info= Nokogiri::XML(Base64::decode64(STDIN.gets.chomp)) for reading from STDIN
api_info = Nokogiri::XML(Base64::decode64(ARGV[0]))

success = api_info.xpath("/CALL_INFO/RESULT").text.to_i == 1
uname   = api_info.xpath('//PARAMETER[TYPE="IN" and POSITION=2]/VALUE').text

if !success
    puts "Failing to create user"
    exit -1
end

puts "User #{uname} successfully created"
```

State Hook example (HOST)

This script prints to stdout the ID of the user that invoked one.host.create API call and the ID of the new host.

```
# Hook template
#
#NAME = hook-error
#TYPE = state
#COMMAND = hook_error.rb
#ARGUMENTS="$TEMPLATE"
#STATE = ERROR
#RESOURCE = HOST

#!/usr/bin/ruby

require 'base64'
require 'nokogiri'

#host_template = Nokogiri::XML(Base64::decode64(STDIN.gets.chomp)) for reading from␣
↪STDIN
host_template = Nokogiri::XML(Base64::decode64(ARGV[0]))

host_id = host_template.xpath("//ID").text.to_i

puts "Host #{host_id} is in error state!!"
```

State Hook example (VM)

```
# Hook template
#
#NAME = vm-prolog
#TYPE = state
#COMMAND = vm_prolog.rb
#ARGUMENTS = $TEMPLATE
#ON = PROLOG
#RESOURCE = VM

#!/usr/bin/ruby

require 'base64'
require 'nokogiri'

#vm_template = Nokogiri::XML(Base64::decode64(STDIN.gets.chomp)) for reading from␣
↪STDIN
vm_template = Nokogiri::XML(Base64::decode64(ARGV[0]))

vm_id = vm_template.xpath("//ID").text.to_i

puts "VM #{vm_id} is in PROLOG state"
```

Note: Note that any command can be specified in `COMMAND`, for debugging. (`COMMAND="/usr/bin/echo"`) can be very helpful.

2.3 Hook Manager Events

The Hook Manager publish the OpenNebula events over a ZeroMQ publisher socket. This can be used for developing custom components that subscribe to specific events to perform custom actions.

2.3.1 Hook Manager Messages

The hook manager sends two different types of messages:

- **EVENT** messages: represent an OpenNebula event, which potentially could trigger a hook if the Hook Execution Manager is subscribed to it.
- **RETRY** messages: represent a retry action of a given hook execution. These event messages are specific to the Hook Execution Manager.

Both messages have a **key** that can be used by the subscriber to receive messages related to an specific event class; and a **content** that contains the information related to the event **encoded in base64**.

Event Messages Format

There are two different types of EVENT messages, representing API and state events. The key format for both types are listed below:

EVENT	Key format
API	`EVENT API <API_CALL> <SUCCESS>` (e.g. `EVENT API one.vm.allocate 1` or `Key: EVENT API one.hook.info 0`)
STATE (HOST)	`EVENT STATE HOST/<STATE>/` (e.g. `EVENT STATE HOST/INIT/`) `EVENT HOST <HOST_ID>/<STATE>/` (e.g. `EVENT HOST 0/INIT/`)
STATE (VM)	`EVENT STATE VM/<STATE>/<LCM_STATE>` (e.g. `EVENT STATE VM/PENDING/LCM_INIT`) `EVENT VM <VM_ID>/<STATE>/<LCM_STATE>` (e.g. `EVENT VM 0/PENDING/LCM_INIT`)

Keys are used to subscribe to specific events. Note also that you do not need to specify the whole key, form example `EVENT STATE HOST/ERROR/` will suscribe for state changes to `ERROR` on the hosts, while `EVENT STATE HOST/` will suscribe for all state changes of the hosts.

The content of an event message is an XML document containing information related to the event, the XSD representing this content are available here.

Retry Messages Format

The key format of the retry messages is just the word `RETRY`, no more info is needed in the key. The retry messages content is an XML file with information about the execution to retry. The XSD representing this content is available here.

2.3.2 Subscriber script example

Below there is an example of script which retrieve all the event messages and prints their key and their content.

```
#!/usr/bin/ruby

require 'ffi-rzmq'
require 'base64'

@context    = ZMQ::Context.new(1)
@subscriber = @context.socket(ZMQ::SUB)

@subscriber.setsockopt(ZMQ::SUBSCRIBE, "EVENT")
@subscriber.connect("tcp://localhost:2101")

key = ''
content = ''

loop do
    @subscriber.recv_string(key)
    @subscriber.recv_string(content)

    puts "Key: #{key}\nContent: #{Base64.decode64(content)}"
    puts "\n================================================================================\n"
end
```

Note: Note that the key and the content as read separately as ZeroMQ envelopes are used.

2.4 Virtualization Driver

The component that deals with the hypervisor to create, manage and get information about virtual machine objects is called Virtual Machine Manager (VMM for short). This component has two parts. The first one resides in the core and holds most of the general functionality common to all the drivers (and some specific), the second is the driver that is the one able to translate basic VMM actions to the hypervisor.

2.4.1 Driver Configuration

There are two main drivers `one_vmm_exec` and `one_vmm_sh`. Both take commands from OpenNebula and execute a set of scripts for those actions, the main difference is that `one_vmm_exec` executes the commands remotely (logging into the host where VM is being or is going to be executed) and `one_vmm_sh` does the execution of the scripts in the frontend.

The driver takes some parameters, described here:

parameter	description
-r <num>	number of retries when executing an action
-t <num	number of threads, i.e. number of actions done at the same time
-l <actions>	(`one_vmm_exec` only) actions executed locally, command can be overridden for each action
<driver_directory>	where in the remotes directory the driver can find the action scripts

These are the actions valid in the -l parameter:

- attach_disk
- attach_nic
- cancel
- deploy
- detach_disk
- detach_nic
- migrate
- migrate_local
- poll
- reboot
- reset
- restore
- save
- shutdown
- snapshot_create
- snapshot_delete
- snapshot_revert

You can also provide an alternative script name for local execution, by default the script is called the same as the action, in this fashion `action=script_name`. As an example:

```
-l migrate,poll=poll_ganglia,save
```

These arguments are specified in the oned.conf file, `arguments` variable:

```
VM_MAD = [
    name       = "kvm",
    executable = "one_vmm_exec",
    arguments  = "-t 15 -r 0 -l migrate,save kvm",
    default    = "vmm_exec/vmm_exec_kvm.conf",
    ...
```

Each driver can define a list of supported actions for imported VMs. Please note that in order to import VMs, your monitoring drivers should report the *IMPORT_TEMPLATE variable*. The complete list of actions is:

- migrate
- live-migrate
- shutdown
- shutdown-hard
- undeploy
- undeploy-hard
- hold
- release
- stop
- suspend
- resume
- delete
- delete-recreate
- reboot
- reboot-hard
- resched
- unresched
- poweroff
- poweroff-hard
- disk-attach
- disk-detach
- nic-attach
- nic-detach
- snap-create
- snap-delete

These supported action are specified in the oned.conf file, `imported_vms_actions` variable:

```
VM_MAD = [
    sunstone_name = "KVM",
    name           = "kvm",
    executable     = "one_vmm_exec",
    arguments      = "-t 15 -r 0 -i kvm",
    default        = "vmm_exec/vmm_exec_kvm.conf",
    type           = "kvm",
    keep_snapshots = "no",
    imported_vms_actions = "shutdown,shutdown-hard,hold,release,suspend,resume,delete,
↪reboot,reboot-hard,resched,unresched,disk-attach,disk-detach,nic-attach,nic-detach,
↪snap-create,snap-delete"
]
```

The hypervisor may preserve system snapshots across power on/off cycles and live migrations, in that case you can set `keep_snapshots` variable to `yes`.

The sunstone name will be used in the host creation dialog in the Sunstone WebUI.

2.4.2 Actions

Every action should have an executable program (mainly scripts) located in the remote dir (`remotes/vmm/<driver_directory>`) that performs the desired action. These scripts receive some parameters (and in the case of `DEPLOY` also STDIN) and give back the error message or information in some cases writing to STDOUT.

VMM actions, they are the same as the names of the scripts:

- **attach_disk**: Attaches a new DISK in the VM
  - Arguments
    - **DOMAIN**: Domain name: one-101
    - **SOURCE**: Image path
    - **TARGET**: Device in the guest: hda, sdc, vda, xvdc
    - **TARGET_INDEX**: Position in the list of disks
    - **DRV_ACTION**: action xml. Base: `/VMM_DRIVER_ACTION_DATA/VM/TEMPLATE/DISK[ATTACH='YES']`
      - `DRIVER`: Disk format: raw, qcow2
      - `TYPE`: Disk type: block, cdrom, rbd, fs or swap
      - `READONLY`: The value is `YES` when it's read only
      - `CACHE`: Cache mode: none, writethrough, writeback
      - `SOURCE`: Image source, used for ceph
  - Response
    - Success: -
    - Failure: Error message
- **attach_nic**: Attaches a new NIC in the VM
  - Arguments
    - **DOMAIN**: Domain name: one-808
    - **MAC**: MAC address of the new NIC

        - **BRIDGE**: Bridge where to attach the new NIC
        - **MODEL**: NIC model to emulate, ex: `e1000`
        - **NET_DRV**: Network driver used, ex: `ovswitch`
        - **TARGET**: Names the VM interface in the host bridge
    - Response
        - Success: -
        - Failure: Error message
- **cancel**: Destroy a VM
    - Arguments:
        - **DOMAIN**: Domain name: one-909
    - Response
        - Success: -
        - Failure: Error message
- **deploy**: Deploy a new VM
    - Arguments:
        - **DEPLOYMENT_FILE**: where to write the deployment file. You have to write whatever comes from STDIN to a file named like this parameter. In shell script you can do: `cat > $domain`
    - Response
        - Success: Deploy id, ex: one-303
        - Failure: Error message
- **detach_disk**: Detaches a DISK from a VM
    - Arguments
        - **DOMAIN**: Domain name: one-286
        - **SOURCE**: Image path
        - **TARGET**: Device in the guest: hda, sdc, vda, xvdc
        - **TARGET_INDEX**: Position in the list of disks
    - Response
        - Success: -
        - Failure: Error message
- **detach_nic**: Detaches a NIC from a VM
    - Arguments
        - **DOMAIN**: Domain name: one-286
        - **MAC**: MAC address of the NIC to detach
    - Response
        - Success: -
        - Failure: Error message

- **migrate**: Live migrate a VM to another host
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **DESTINATION_HOST**: Host where to migrate the VM
    - **HOST**: Host where the VM is currently running
  - Response
    - Success: -
    - Failure: Error message
- **migrate_local**: Live migrate a VM to another host, initiating the connection from the front-end
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **DESTINATION_HOST**: Host where to migrate the VM
    - **HOST**: Host where the VM is currently running
  - Response
    - Success: -
    - Failure: Error message
- **poll**: Get information from a VM
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **HOST**: Host where the VM is running
  - Response
    - Success: -
    - Failure: Error message
- **reboot**: Orderly reboots a VM
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **HOST**: Host where the VM is running
  - Response
    - Success: -
    - Failure: Error message
- **reset**: Hard reboots a VM
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **HOST**: Host where the VM is running
  - Response
    - Success: -

        * Failure: Error message

- **restore**: Restores a previously saved VM
  - Arguments:
    * **FILE**: VM save file
    * **HOST**: Host where to restore the VM
  - Response
    * Success: -
    * Failure: Error message
- **restore.<SYSTEM_TM>**: *[Only for KVM drivers]* If this script exists, the `restore` script will execute it right at the beginning to extract the checkpoint from the system datastore. For example, for the `ceph` system datastore the `restore.ceph` script is defined.
  - Arguments:
    * **FILE**: VM save file
    * **HOST**: Host where to restore the VM
- **save**: Saves a VM
  - Arguments:
    * **DOMAIN**: Domain name: one-286
    * **FILE**: VM save file
    * **HOST**: Host where the VM is running
  - Response
    * Success: -
    * Failure: Error message
- **save.<SYSTEM_TM>**: *[Only for KVM drivers]* If this script exists, the `save` script will execute it right at the end to store the checkpoint in the system datastore. For example, for the `ceph` system datastore the `save.ceph` script is defined.
  - Arguments:
    * **DOMAIN**: Domain name: one-286
    * **FILE**: VM save file
    * **HOST**: Host where the VM is running
- **shutdown**: Orderly shutdowns a VM
  - Arguments:
    * **DOMAIN**: Domain name: one-286
    * **HOST**: Host where the VM is running
  - Response
    * Success: -
    * Failure: Error message
- **snapshot_create**: Makes a new snapshot of a VM

  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **ONE_SNAPSHOT_ID**: OpenNebula snapshot identifier
  - Response
    - Success: Snapshot name for the hypervisor. Used later to delete or revert
    - Failure: Error message
- **snapshot_delete**: Deletes a snapshot of a VM
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **SNAPSHOT_NAME**: Name used to refer the snapshot in the hypervisor
  - Response
    - Success: -
    - Failure: Error message
- **snapshot_revert**: Returns a VM to an saved state
  - Arguments:
    - **DOMAIN**: Domain name: one-286
    - **SNAPSHOT_NAME**: Name used to refer the snapshot in the hypervisor
  - Response
    - Success: -
    - Failure: Error message

`action xml` parameter is a base64 encoded xml that holds information about the VM. To get one of the values explained in the documentation, for example from `attach_disk` `READONLY` you can add to the base XPATH the name of the parameter. XPATH:

```
/VMM_DRIVER_ACTION_DATA/VM/TEMPLATE/DISK[ATTACH='YES']/READONLY
```

When using shell script there is a handy script that gets parameters for given XPATH in that XML. Example:

```
XPATH="${DRIVER_PATH}/../../datastore/xpath.rb -b $DRV_ACTION"

unset i j XPATH_ELEMENTS

DISK_XPATH="/VMM_DRIVER_ACTION_DATA/VM/TEMPLATE/DISK[ATTACH='YES']"

while IFS= read -r -d '' element; do
    XPATH_ELEMENTS[i++]="$element"
done < <($XPATH     $DISK_XPATH/DRIVER \
                    $DISK_XPATH/TYPE \
                    $DISK_XPATH/READONLY \
                    $DISK_XPATH/CACHE \
                    $DISK_XPATH/SOURCE)

DRIVER="${XPATH_ELEMENTS[j++]:-$DEFAULT_TYPE}"
TYPE="${XPATH_ELEMENTS[j++]}"
READONLY="${XPATH_ELEMENTS[j++]}"
```

(continues on next page)

(continued from previous page)

```
CACHE="${XPATH_ELEMENTS[j++]}"
IMG_SRC="${XPATH_ELEMENTS[j++]}"
```

`one_vmm_sh` has the same script actions and meanings but an argument more that is the host where the action is going to be performed. This argument is always the first one. If you use `-p` parameter in `one_vmm_ssh` the poll action script is called with one more argument that is the host where the VM resides, also it is the same parameter.

2.4.3 Poll Information

`POLL` is the action that gets monitoring info from the running VMs. This action is called when the VM is not found in the host monitoring process for whatever reason. The format it is supposed to give back information is a line with `KEY=VALUE` pairs separated by spaces. It also supports vector values `KEY = [ SK1=VAL1, SK2=VAL2 ]`. An example monitoring output looks like this:

```
STATE=a USEDMEMORY=554632 DISK_SIZE=[ ID=0, SIZE=24 ] DISK_SIZE=[ ID=1, SIZE=242 ]␣
↪SNAPSHOT_SIZE=[ ID=0, DISK_ID=0, SIZE=24 ]
```

The poll action can give back any information and it will be added to the VM information hold but there are some variables that should be given back as they are meaningful to OpenNebula:

Variable	Description
STATE	State of the VM (explained later)
CPU	Percentage of 1 CPU consumed (two fully consumed cpu is 200)
MEMORY	Memory consumption in kilobytes
NETRX	Received bytes from the network
NETTX	Sent bytes to the network
DISKRD-BYTES	Read bytes from all disks since last VM start
DISKWR-BYTES	Written bytes from all disks since last VM start
DISKRDIOPS	Read IO operations from all disks since last VM start
DISKWRIOPS	Written IO operations all disks since last VM start
DISK_SIZE	Vector attribute two sub-attributes: `ID` id of the disk, and `SIZE` real size of the disk in MB
SNAP-SHOT_SIZE	Vector attribute two sub-attributes: `ID` id of the snapshot, `DISK_ID` id of the disk, and `SIZE` real size of the snapshot in MB

`STATE` is a single character that tells OpenNebula the status of the VM, the states are the ones in this table:

state	description
N/A	Detecting state error. The monitoring could be done, but it returned an unexpected output.
a	Active. The VM alive (running, blocked, booting...). The VM will be set to `RUNNING`
p	Paused. The VM will be set to `SUSPENDED`
e	Error. The VM crashed or somehow its deployment failed. The VM will be set to `UNKNOWN`
d	Disappeared. VM not known by the hypervisor anymore. The VM will be set to `POWEROFF`

2.4.4 Deployment File

The deployment file is a text file written by OpenNebula core that holds the information of a VM. It is used when deploying a new VM. OpenNebula is able to generate three formats of deployment files:

- **kvm**: libvirt format used to create kvm VMs
- **xml**: xml representation of the VM

If the target hypervisor is not libvirt/kvm the best format to use is xml as it holds more information than the two others. It has all the template information encoded as xml. This is an example:

```
<TEMPLATE>
  <CPU><![CDATA[1.0]]></CPU>
  <DISK>
    <DISK_ID><![CDATA[0]]></DISK_ID>
    <SOURCE><![CDATA[/home/user/vm.img]]></SOURCE>
    <TARGET><![CDATA[sda]]></TARGET>
  </DISK>
  <MEMORY><![CDATA[512]]></MEMORY>
  <NAME><![CDATA[test]]></NAME>
  <VMID><![CDATA[0]]></VMID>
</TEMPLATE>
```

There are some information added by OpenNebula itself like the VMID and the `DISK_ID` for each disk. `DISK_ID` is very important as the disk images are previously manipulated by the `TM` driver and the disk should be accessible at `VM_DIR/VMID/images/disk.DISK_ID`.

2.5 Provision Driver

The provision driver communicates with the remote infrastructure provider (e.g. bare metal cloud hosting) to allocate and control new resources. These resources are expected to be later added into the OpenNebula as virtualization hosts, and can represent new individual bare metal hosts (e.g., to act as KVM hypervisors) or VMware vCenter.

The OpenNebula server isn't dealing directly with the provision drivers. Standalone tool `oneprovision` provides full interaction between the drivers (triggering particular driver actions) and the OpenNebula (creating and managing provisioned host objects).

The structure of the driver with actions is very similar to the *Virtualization Driver*, the main difference is the type of objects the drivers deal with. The virtualization driver works with the OpenNebula VMs, but the provision driver works with the OpenNebula hosts.

2.5.1 Actions

Every action should have an executable program (mainly scripts) located in the remote dir (`remotes/pm/<driver_directory>`) that performs the desired action. These scripts receive some parameters (and in the case of `DEPLOY` also STDIN) and give back the error message or information in some cases writing to STDOUT.

Note: Except the listed arguments, all action scripts also get following arguments after all specified arguments:

- **HOST_ID** - ID of the OpenNebula host
- **HOST** - Name of the OpenNebula host

Provision actions, they are the same as the names of the scripts:

- **cancel**: Destroy a provision
  - Arguments:
    - **DEPLOY_ID**: Provision deployment ID

        * **HOST**: Name of OpenNebula host
    - Response
        * Success: -
        * Failure: Error message
- **deploy**: Create new provision
    - Arguments:
        * **DEPLOYMENT_FILE**: where to write the deployment file. You have to write whatever comes from STDIN to a file named like this parameter. In shell script you can do: `cat > $domain`
        * **HOST**: Name of OpenNebula host
    - Response
        * Success: Deploy ID, unique identification from provider
        * Failure: Error message
- **poll**: Get information about a provisioned host
    - Arguments:
        * **DEPLOY_ID**: Provision deployment ID
        * **HOST**: Name of OpenNebula host
    - Response
        * Success: Output as for **poll** action in the *Virtualization Driver*
        * Failure: Error message
- **reboot**: Orderly reboots a provisioned host
    - Arguments:
        * **DEPLOY_ID**: Provision deployment ID
        * **HOST**: Name of OpenNebula host
    - Response
        * Success: -
        * Failure: Error message
- **reset**: Hard reboots a provisioned host
    - Arguments:
        * **DEPLOY_ID**: Provision deployment ID
        * **HOST**: Name of OpenNebula host
    - Response
        * Success: -
        * Failure: Error message
- **shutdown**: Orderly shutdowns a provisioned host
    - Arguments:
        * **DEPLOY_ID**: Provision deployment ID

        * **HOST**: Name of OpenNebula host
        * **LCM_STATE**: Emulated LCM_STATE string
    - Response
        * Success: -
        * Failure: Error message

2.5.2 Deployment File

The deployment file contains the OpenNebula host XML object representation. It's may be used in the situation when OpenNebula host still isn't in the database and doesn't have host ID assigned, but the driver actions need to work with its metadata.

Example:

```
<HOST>
  <NAME>provision-c968cbcf40716f8e18caddbb8757c2ca7ed0942a357d511b</NAME>
  <IM_MAD>kvm</IM_MAD>
  <VM_MAD>kvm</VM_MAD>
  <PM_MAD>packet</PM_MAD>
  <TEMPLATE>
    <IM_MAD>kvm</IM_MAD>
    <VM_MAD>kvm</VM_MAD>
    <PM_MAD>packet</PM_MAD>
    <PROVISION>
      <PACKET_TOKEN>*****</PACKET_TOKEN>
      <PACKET_PROJECT>*****</PROJECT>
      <FACILITY>ams1</FACILITY>
      <PLAN>baremetal_0</PLAN>
      <HOSTNAME>TestOneProvision1-C7</HOSTNAME>
      <BILLING_CYCLE>hourly</BILLING_CYCLE>
      <OS>centos_7</OS>
    </PROVISION>
    <PROVISION_CONFIGURATION_BASE64>*****</PROVISION_CONFIGURATION_BASE64>
    <PROVISION_CONFIGURATION_STATUS>pending</PROVISION_CONFIGURATION_STATUS>
    <PROVISION_CONNECTION>
      <REMOTE_USER>root</REMOTE_USER>
      <REMOTE_PORT>22</REMOTE_PORT>
      <PUBLIC_KEY>/var/lib/one/.ssh/ddc/id_rsa.pub</PUBLIC_KEY>
      <PRIVATE_KEY>/var/lib/one/.ssh/ddc/id_rsa</PRIVATE_KEY>
    </PROVISION_CONNECTION>
    <CONTEXT>
      <SSH_PUBLIC_KEY>*****</SSH_PUBLIC_KEY>
    </CONTEXT>
  </TEMPLATE>
</HOST>
```

2.6 Storage Driver

The Storage subsystem is highly modular. These drivers are separated into two logical sets:

- **DS**: Datastore drivers. They serve the purpose of managing images: register, delete, and create empty datablocks.

- **TM**: Transfer Manager drivers. They manage images associated to instantiated VMs.

2.6.1 Datastore Drivers Structure

Located under `/var/lib/one/remotes/datastore/<ds_mad>`

- **cp**: copies/dumps the image to the datastore
  - **ARGUMENTS**: `datastore_image_dump image_id`
  - **RETURNS**: `image_source size`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
  - `image_source` is the image source which will be later sent to the transfer manager
- **mkfs**: creates a new empty image in the datastore
  - **ARGUMENTS**: `datastore_image_dump image_id`
  - **RETURNS**: `image_source size`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
  - `image_source` is the image source which will be later sent to the transfer manager.
- **rm**: removes an image from the datastore
  - **ARGUMENTS**: `datastore_image_dump image_id`
  - **RETURNS**: –
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
- **stat**: returns the size of an image in Mb
  - **ARGUMENTS**: `datastore_image_dump image_id`
  - **RETURNS**: `size`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
  - `size` the size of the image in Mb.
- **clone**: clones an image.
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: `source`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
  - `source` the new `source` for the image.
- **monitor**: monitors a datastore
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: `monitor data`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.

  - `monitor data` The monitoring information of the datastore, namely "USED_MB=...\nTOTAL_MB=...\nFREE_MB=..." which are respectively the used size of the datastore in MB, the total capacity of the datastore in MB and the available space in the datastore in MB.

- **snap_delete**: Deletes a snapshot of a persistent image.
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: –
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*. This dump, in addition to the elements in the example, contains a ROOT element: `TARGET_SNAPSHOT`, with the ID of the snapshot.
- **snap_flatten**: Flattens a snapshot. The operation results in an image without snapshots, with the contents of the selected snapshot.
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: –
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*. This dump, in addition to the elements in the example, contains a ROOT element: `TARGET_SNAPSHOT`, with the ID of the snapshot.
- **snap_revert**: Overwrites the contents of the image by the selected snapshot (discarding any non-saved changes).
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: –
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*. This dump, in addition to the elements in the example, contains a ROOT element: `TARGET_SNAPSHOT`, with the ID of the snapshot.
- **export**: Generates an XML file required to export an image from a datastore. This script represents only the first part of the export process, it only generates metadata (an xml). The information returned by this script is then fed to `downloader.sh` which completes the export process.
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: `export_xml`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*. This dump, in addition to the elements in the example, contains a ROOT element: `TARGET_SNAPSHOT`, with the ID of the snapshot.
  - `export_xml`: The XML response should follow *this* structure. The variables that appear within are the following:
    * `<src>`: The SOURCE of the image (path to the image in the datastore)
    * `<md5>`: The MD5 of the image
    * `<format>`: The format of the image, e.g.: `qcow2`, `raw`, `vmdk`, `unknown`...
    * `<dispose>`: Can be either `YES` or `NO`. Dispose will remove the image from the reported path after the image has been successfully exported. This is regularly not necessary if the `downloader.sh` script can access the path to the image directly in the datastore (`src`).

Note: `image_source` has to be dynamically generated by the `cp` and `mkfs` script. It will be passed later on to the transfer manager, so it should provide all the information the transfer manager needs to locate the image.

2.6.2 TM Drivers Structure

This is a list of the TM drivers and their action. Note that they don't return anything. If the exit code is not `0`, the driver failed.

Located under `/var/lib/one/remotes/tm/<tm_mad>`. There are two types of action scripts: the first group applies to general image datastores and includes (`clone`, `ln`, `mv` and `mvds`); the second one is only used in conjunction with the system datastore.

Action scripts for generic image datastores:

- **clone**: clones the image from the datastore (non-persistent images)
  - **ARGUMENTS**: `fe:SOURCE host:remote_system_ds/disk.i vm_id ds_id`
  - `fe` is the front-end hostname
  - `SOURCE` is the path of the disk image in the form DS_BASE_PATH/disk
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **ln**: Links the image from the datastore (persistent images)
  - **ARGUMENTS**: `fe:SOURCE host:remote_system_ds/disk.i vm_id ds_id`
  - `fe` is the front-end hostname
  - `SOURCE` is the path of the disk image in the form DS_BASE_PATH/disk
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **mvds**: moves an image back to its datastore (persistent images)
  - **ARGUMENTS**: `host:remote_system_ds/disk.i fe:SOURCE vm_id ds_id`
  - `fe` is the front-end hostname
  - `SOURCE` is the path of the disk image in the form DS_BASE_PATH/disk
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the original datastore for the image)
- **cpds**: copies an image back to its datastore (executed for the saveas operation)
  - **ARGUMENTS**: `host:remote_system_ds/disk.i fe:SOURCE snap_id vm_id ds_id`
  - `fe` is the front-end hostname
  - `SOURCE` is the path of the disk image in the form DS_BASE_PATH/disk
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host

  - `snap_id` the ID of the snapshot to save. If the ID is -1 it saves the current state and not a snapshot.
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the original datastore for the image)

- **mv**: moves images/directories across system_ds in different hosts. When used for the system datastore the script will received the directory ARGUMENT. This script will be also called for the image TM for each disk to perform setup tasks on the target node.
  - **ARGUMENTS**: `hostA:system_ds/disk.i|hostB:system_ds/disk.i vm_id ds_id` OR `hostA:system_ds/|hostB:system_ds/ vm_id ds_id`
  - `hostA` is the host the VM is in.
  - `hostB` is the target host to deploy the VM
  - `system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)

Note: You only need to implement one `mv` script, but consider the arguments received when the TM is used for the system datastore, a regular image datastore or both.

- **premigrate**: It is executed before a livemigration operation is issued to the hypervisor. Note that **only the premigrate script from the system datastore will be used**. Any customization must be done for the premigrate script of the system datastore, although you will probably add operations for other backends than that used by the system datastore.
  - **ARGUMENTS**: `source_host dst_host remote_system_dir vmid dsid template`
  - `src_host` is the host the VM is in.
  - `dst_host` is the target host to migrate the VM to
  - `remote_system_ds_dir` is the path for the VM directory in the system datastore in the host
  - `vmid` is the id of the VM
  - `dsid` is the target datastore
  - `template` is the template of the VM in XML and base64 encoded
- **postmigrate**: It is executed after a livemigration operation. Note that **only the postmigrate script from the system datastore will be used**. Any customization must be done for the postmigrate script of the system datastore, although you will probably add operations for other backends than that used by the system datastore.
  - **ARGUMENTS**: `source_host dst_host remote_system_dir vmid dsid template`
  - see `premigrate` description.
- **snap_create**: Creates a disk snapshot of the selected disk
  - **ARGUMENTS**: `host:remote_system_ds/disk.i snapshot_id vm_id ds_id`
  - `remote_system_ds_dir` is the path for the VM directory in the system datastore in the host
  - `host` is the target host where the VM is running
  - `snapshot_id` the id of the snapshot to be created/reverted to/deleted
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)

- **snap_create_live**: Creates a disk snapshot of the selected disk while the VM is running in the hypervisor. This is a hypervisor operation.
  - **ARGUMENTS**: `host:remote_system_ds/disk.i snapshot_id vm_id ds_id`
  - `remote_system_ds_dir` is the path for the VM directory in the system datastore in the host
  - `host` is the target host where the VM is running
  - `snapshot_id` the id of the snapshot to be created/reverted to/deleted
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **snap_delete**: Deletes a disk snapshot
  - **ARGUMENTS**: `host:remote_system_ds/disk.i snapshot_id vm_id ds_id`
  - see `snap_create` description.
- **snap_revert**: Reverts to the selected snapshot (and discards any changes to the current disk)
  - **ARGUMENTS**: `host:remote_system_ds/disk.i snapshot_id vm_id ds_id`
  - see `snap_create` description.

Action scripts needed when the TM is used for the system datastore:

- **context**: creates an ISO that contains all the files passed as an argument.
  - **ARGUMENTS**: `file1 file2 ... fileN host:remote_system_ds/disk.i vm_id ds_id`
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **delete**: removes the either system datastore's directory of the VM or a disk itself.
  - **ARGUMENTS**: `host:remote_system_ds/disk.i|host:remote_system_ds/ vm_id ds_id`
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **mkimage**: creates an image on-the-fly bypassing the datastore/image workflow
  - **ARGUMENTS**: `size format host:remote_system_ds/disk.i vm_id ds_id`
  - `size` size in MB of the image
  - `format` format for the image
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)

- **mkswap**: creates a swap image
  - **ARGUMENTS**: `size host:remote_system_ds/disk.i vm_id ds_id`
  - `size` size in MB of the image
  - `host` is the target host to deploy the VM
  - `remote_system_ds` is the path for the system datastore in the host
  - `vm_id` is the id of the VM
  - `ds_id` is the target datastore (the system datastore)
- **monitor**: monitors a **shared** system datastore. Non-shared system datastores are monitored through `monitor_ds` script.
  - **ARGUMENTS**: `datastore_action_dump image_id`
  - **RETURNS**: `monitor data`
  - `datastore_image_dump` is an XML dump of the driver action encoded in Base 64. See a decoded *example*.
  - `monitor data` Including:
    - The monitoring information of the datastore, namely "USED_MB=...\nTOTAL_MB=...\nFREE_MB=..." which are respectively the used size of the datastore in MB, the total capacity of the datastore in MB and the available space in the datastore in MB.
    - It also needs to return for each VM the size of each disk and any snapshot on those disks. In the form:

```
VM = [ ID = ${vm_id}, POLL = "\
    DISK_SIZE=[ID=${disk_id},SIZE=${disk_size}]
    ...
    SNAPSHOT_SIZE=[ID=${snap},DISK_ID=${disk_id},SIZE=${snap_size}]
    ...
    "
]
...
```

- **monitor_ds**: monitors a **ssh-like** system datastore. Distributed system datastores should `exit 0` on the previous monitor script. Arguments and return values are the same as the monitor script.

Note: If the TM is only for regular images you only need to implement the first group.

2.6.3 Tunning OpenNebula Core and Driver Integration

The behavior of OpenNebula can be adapted depending on how the storage perform the underlying operations. For example quotas are computed on the original image datastore depending on the CLONE attribute. In particular, you may want to set two configuration attributes for your drivers: `DS_MAD_CONF` and `TM_MAD_CONF`. See the OpenNebula configuration reference for details.

2.6.4 The Monitoring Process

Image Datastores

The information is obtained periodically using the Datastore driver monitor script

Shared System Datastores

These datastores are monitored from a single point once (either the front-end or one of the storage bridges in `BRIDGE_LIST`). This will prevent overloading the storage by all the nodes querying it at the same time.

The driver plugin `<tm_mad>/monitor` will report the information for two things:

- Total storage metrics for the datastore (`USED_MB` `FREE_MB` `TOTAL_MB`)
- Disk usage metrics (all disks: volatile, persistent and non-persistent)

Non-shared System Datastores (SSH-like)

Non-shared SSH datastores are labeled by including a `.monitor` file in the datastore directory in any of the clone or ln operations. Only those datastores are monitored remotely by the monitor_ds.sh probe. The datastore is monitored with `<tm_mad>/monitor_ds`, but `tm_mad` is obtained by the probes reading from the .monitor file.

The plugins <tm_mad>/monitor_ds + kvm-probes.d/monitor_ds.sh will report the information for two things:

- Total storage metrics for the datastore (`USED_MB` `FREE_MB` `TOTAL_MB`)
- Disk usage metrics (all disks volatile, persistent and non-persistent)

Note: `.monitor` will be only present in SSH datastores to be monitored in the nodes. System Datastores that need to be monitored in the nodes will need to provide a `monitor_ds` script and not the `monitor` one. This is to prevent errors, and not invoke the shared mechanism for local datastores.

The monitor_ds script.

The monitor_ds.sh probe from the IM, if the `.monitor` file is present (e.g. `/var/lib/one/datastores/100/.monitor`), will execute its contents in the form `/var/tmp/one/remotes/tm/$(cat .monitor)/monitor_ds /var/lib/one/datastores/100/`. Note that the argument is the datastore path and not the VM or VM disk.

The script is responsible from getting the information from all disks of all VMs in the datastore in that node.

2.6.5 Mixed Transfer modes

Certain types of TM can be used in so called *mixed mode* and allow different types of image and system datastore drivers to work together.

The following combinations are supported by default:

- **CEPH + SSH** described in Ceph SSH mode
- **Qcow2/shared + SSH** described in Qcow2/shared SSH mode

The support in oned is generic, in a *mixed mode* every TM action (such as `clone` or `delete`) is suffixed with the driver name of the system DS in use. For example, an action like `clone.ssh` is actually invoked in CEPH + SSH mode. The driver first tries to find the complete action script, including the system DS suffix (e.g. `ceph/clone.ssh`) and only if that does not exist fallbacks to the default (`ceph/clone`).

In this way other combinations can be easily added.

2.6.6 An Example VM

Consider a VM with two disks:

```
NAME   = vm01
CPU    = 0.1
MEMORY = 64

DISK   = [ IMAGE_ID = 0 ] # non-persistent disk
DISK   = [ IMAGE_ID = 1 ] # persistent disk
```

This a list of TM actions that will be called upon the events listed:

CREATE

```
<tm_mad>/clone <frontend>:<non_pers_image_source> <host01>:<ds_path>/<vm_id>/disk.0
<tm_mad>/ln <frontend>:<pers_image_source> <host01>:<ds_path>/<vm_id>/disk.1
```

STOP

```
<tm_mad>/mv <host01>:<ds_path>/<vm_id>/disk.0 <frontend>:<ds_path>/<vm_id>/disk.0
<tm_mad>/mv <host01>:<ds_path>/<vm_id>/disk.1 <frontend>:<ds_path>/<vm_id>/disk.1
<tm_mad_sysds>/mv <host01>:<ds_path>/<vm_id> <frontend>:<ds_path>/<vm_id>
```

RESUME

```
<tm_mad>/mv <frontend>:<ds_path>/<vm_id>/disk.0 <host01>:<ds_path>/<vm_id>/disk.0
<tm_mad>/mv <frontend>:<ds_path>/<vm_id>/disk.1 <host01>:<ds_path>/<vm_id>/disk.1
<tm_mad_sysds>/mv <frontend>:<ds_path>/<vm_id> <host01>:<ds_path>/<vm_id>
```

MIGRATE host01 → host02

```
<tm_mad>/mv <host01>:<ds_path>/<vm_id>/disk.0 <host02>:<ds_path>/<vm_id>/disk.0
<tm_mad>/mv <host01>:<ds_path>/<vm_id>/disk.1 <host02>:<ds_path>/<vm_id>/disk.1
<tm_mad_sysds>/mv <host01>:<ds_path>/<vm_id> <host02>:<ds_path>/<vm_id>
```

SHUTDOWN

```
<tm_mad>/delete <host02>:<ds_path>/<vm_id>/disk.0
<tm_mad>/mvds <host02>:<ds_path>/<vm_id>/disk.1 <pers_image_source>
<tm_mad_sysds>/delete <host02>:<ds_path>/<vm_id>
```

- `non_pers_image_source`: Source of the non persistent image.
- `pers_image_source` : Source of the persistent image.
- `frontend`: hostname of the frontend
- `host01`: hostname of host01
- `host02`: hostname of host02
- `tm_mad`: TM driver of the datastore where the image is registered
- `tm_mad_sysds`: TM driver of the system datastore

2.6.7 Helper Scripts

There is a helper shell script with some functions defined to do some common tasks. It is located in `/var/lib/one/remotes/scripts_common.sh`

Here are the description of those functions.

- **log**: Takes one parameter that is a message that will be logged into the VM log file.

```
log "Creating directory $DST_DIR"
```

- **error_message**: sends an exit message to oned surrounding it by separators, use to send the error message when a command fails.

```
error_message "File '$FILE' not found"
```

- **arg_host**: gets the hostname part from a parameter

```
SRC_HOST=`arg_host $SRC`
```

- **arg_path**: gets the path part from a parameter

```
SRC_PATH=`arg_path $SRC`
```

- **exec_and_log**: executes a command and logs its execution. If the command fails the error message is sent to oned and the script ends

```
exec_and_log "chmod g+w $DST_PATH"
```

- **ssh_exec_and_log**: This function executes $2 at $1 host and report error $3

```
ssh_exec_and_log "$HOST" "chmod g+w $DST_PATH" "Error message"
```

- **timeout_exec_and_log**: like `exec_and_log` but takes as first parameter the max number of seconds the command can run

```
timeout_exec_and_log 15 "cp $SRC_PATH $DST_PATH"
```

The are additional minor helper functions, please read the `scripts_common.sh` to see them.

2.6.8 Decoded Example

```
<DS_DRIVER_ACTION_DATA>
    <IMAGE>
        <ID>0</ID>
        <UID>0</UID>
        <GID>0</GID>
        <UNAME>oneadmin</UNAME>
        <GNAME>oneadmin</GNAME>
        <NAME>ttylinux</NAME>
        <PERMISSIONS>
            <OWNER_U>1</OWNER_U>
            <OWNER_M>1</OWNER_M>
            <OWNER_A>0</OWNER_A>
            <GROUP_U>0</GROUP_U>
            <GROUP_M>0</GROUP_M>
            <GROUP_A>0</GROUP_A>
            <OTHER_U>0</OTHER_U>
            <OTHER_M>0</OTHER_M>
            <OTHER_A>0</OTHER_A>
        </PERMISSIONS>
```

(continues on next page)

(continued from previous page)

```
        <TYPE>0</TYPE>
        <DISK_TYPE>0</DISK_TYPE>
        <PERSISTENT>0</PERSISTENT>
        <REGTIME>1385145541</REGTIME>
        <SOURCE/>
        <PATH>/tmp/ttylinux.img</PATH>
        <FSTYPE/>
        <SIZE>40</SIZE>
        <STATE>4</STATE>
        <RUNNING_VMS>0</RUNNING_VMS>
        <CLONING_OPS>0</CLONING_OPS>
        <CLONING_ID>-1</CLONING_ID>
        <DATASTORE_ID>1</DATASTORE_ID>
        <DATASTORE>default</DATASTORE>
        <VMS/>
        <CLONES/>
        <TEMPLATE>
            <DEV_PREFIX><![CDATA[hd]]></DEV_PREFIX>
            <PUBLIC><![CDATA[YES]]></PUBLIC>
        </TEMPLATE>
    </IMAGE>
    <DATASTORE>
        <ID>1</ID>
        <UID>0</UID>
        <GID>0</GID>
        <UNAME>oneadmin</UNAME>
        <GNAME>oneadmin</GNAME>
        <NAME>default</NAME>
        <PERMISSIONS>
            <OWNER_U>1</OWNER_U>
            <OWNER_M>1</OWNER_M>
            <OWNER_A>0</OWNER_A>
            <GROUP_U>1</GROUP_U>
            <GROUP_M>0</GROUP_M>
            <GROUP_A>0</GROUP_A>
            <OTHER_U>1</OTHER_U>
            <OTHER_M>0</OTHER_M>
            <OTHER_A>0</OTHER_A>
        </PERMISSIONS>
        <DS_MAD>fs</DS_MAD>
        <TM_MAD>shared</TM_MAD>
        <TYPE>0</TYPE>
        <DISK_TYPE>0</DISK_TYPE>
        <CLUSTER_ID>-1</CLUSTER_ID>
        <CLUSTER/>
        <TOTAL_MB>86845</TOTAL_MB>
        <FREE_MB>20777</FREE_MB>
        <USED_MB>1000</USED_MB>
        <IMAGES/>
        <TEMPLATE>
            <CLONE_TARGET><![CDATA[SYSTEM]]></CLONE_TARGET>
            <DISK_TYPE><![CDATA[FILE]]></DISK_TYPE>
            <DS_MAD><![CDATA[fs]]></DS_MAD>
            <LN_TARGET><![CDATA[NONE]]></LN_TARGET>
            <TM_MAD><![CDATA[shared]]></TM_MAD>
            <TYPE><![CDATA[IMAGE_DS]]></TYPE>
        </TEMPLATE>
```

(continues on next page)

(continued from previous page)

```
    </DATASTORE>
</DS_DRIVER_ACTION_DATA>

<DS_DRIVER_ACTION_DATA>
    <DATASTORE>
        <ID>0</ID>
        <UID>0</UID>
        <GID>0</GID>
        <UNAME>oneadmin</UNAME>
        <GNAME>oneadmin</GNAME>
        <NAME>system</NAME>
        <PERMISSIONS>
            <OWNER_U>1</OWNER_U>
            <OWNER_M>1</OWNER_M>
            <OWNER_A>0</OWNER_A>
            <GROUP_U>1</GROUP_U>
            <GROUP_M>0</GROUP_M>
            <GROUP_A>0</GROUP_A>
            <OTHER_U>0</OTHER_U>
            <OTHER_M>0</OTHER_M>
            <OTHER_A>0</OTHER_A>
        </PERMISSIONS>
        <DS_MAD><![CDATA[-]]></DS_MAD>
        <TM_MAD><![CDATA[qcow2]]></TM_MAD>
        <BASE_PATH><![CDATA[/var/lib/one//datastores/0]]></BASE_PATH>
        <TYPE>1</TYPE>
        <DISK_TYPE>0</DISK_TYPE>
        <STATE>0</STATE>
        <CLUSTERS>
            <ID>0</ID>
        </CLUSTERS>
        <TOTAL_MB>31998</TOTAL_MB>
        <FREE_MB>12650</FREE_MB>
        <USED_MB>17694</USED_MB>
        <IMAGES></IMAGES>
        <TEMPLATE>
            <ALLOW_ORPHANS><![CDATA[NO]]></ALLOW_ORPHANS>
            <DS_MIGRATE><![CDATA[YES]]></DS_MIGRATE>
            <SHARED><![CDATA[YES]]></SHARED>
            <TM_MAD><![CDATA[qcow2]]></TM_MAD>
            <TYPE><![CDATA[SYSTEM_DS]]></TYPE>
        </TEMPLATE>
    </DATASTORE>
    <DATASTORE_LOCATION>/var/lib/one//datastores</DATASTORE_LOCATION>
    <MONITOR_VM_DISKS>1</MONITOR_VM_DISKS>
</DS_DRIVER_ACTION_DATA>
```

2.6.9 Export XML

```
<IMPORT_INFO>
   <IMPORT_SOURCE><![CDATA[<src>]]></IMPORT_SOURCE>
   <MD5><![CDATA[<md5sum>]]></MD5>
   <SIZE><![CDATA[<size>]]></SIZE>
   <FORMAT><![CDATA[<format>]></FORMAT>
   <DISPOSE><dispose></DISPOSE>
```

(continues on next page)

(continued from previous page)

```
</IMPORT_INFO>
```

2.7 Monitoring Driver

The Monitoring Drivers (or IM drivers) collect host and virtual machine monitoring data by executing a set of probes in the hosts. This data is either actively queried by OpenNebula or sent periodically by an agent running in the hosts to the frontend.

This guide describes the process of customize or add probes to the hosts. It is also a starting point on how to create a new IM driver from scratch.

2.7.1 Probe Location

The default probes are installed in the frontend in the following path:

- **KVM**: `/var/lib/one/remotes/im/kvm-probes.d`
- **vCenter, EC2 and Azure**: `/var/lib/one/remotes/im/<hypervisor>.d`

In the case of `KVM`, the probes are distributed to the hosts, therefore if the probes are changed, they **must** be distributed to the hosts by running `onehost sync`.

2.7.2 General Probe Structure

An IM driver is composed of one or several scripts that write to `stdout` information in this form:

```
KEY1="value"
KEY2="another value with spaces"
```

The drivers receive the following parameters:

Position	Description
1	**hypervisor**: The name of the hypervisor: `kvm`, etc...
2	**datastore location**: path of the datastores directory in the host
3	**collectd port**: port in which the `collectd` is listening on
4	**monitor push cycle**: time in seconds between monitorization actions for the UDP-push model
5	**host_id**: id of the host
6	**host_name**: name of the host

Take into account that in shell script the parameters start at 1 (`$1`) and in ruby start at 0 (`ARGV[0]`). For shell script you can use this snippet to get the parameters:

```
hypervisor=$1
datastore_location=$2
collectd_port=$3
monitor_push_cycle=$4
host_id=$5
host_name=$6
```

2.7.3 Basic Monitoring Scripts

You can add any key and value you want to use later in `RANK` and `REQUIREMENTS` for scheduling but there are some basic values you should output:

Key	Description
HYPERVISOR	Name of the hypervisor of the host, useful for selecting the hosts with an specific technology.
TOTALCPU	Number of CPUs multiplied by 100. For example, a 16 cores machine will have a value of 1600.
CPUSPEED	Speed in Mhz of the CPUs.
TOTALMEMORY	Maximum memory that could be used for VMs. It is advised to take out the memory used by the hypervisor.
USEDMEMORY	Memory used, in kilobytes.
FREEMEMORY	Available memory for VMs at that moment, in kilobytes.
FREECPU	Percentage of idling CPU multiplied by the number of cores. For example, if 50% of the CPU is idling in a 4 core machine the value will be 200.
USEDCPU	Percentage of used CPU multiplied by the number of cores.
NETRX	Received bytes from the network
NETTX	Transferred bytes to the network

For example, a probe that gets memory information about a host could be something like:

```
#!/bin/bash

total=$(free | awk ' /^Mem/ { print $2 }')
used=$(free | awk '/buffers\/cache/ { print $3 }')
free=$(free | awk '/buffers\/cache/ { print $4 }')

echo "TOTALMEMORY=$total"
echo "USEDMEMORY=$used"
echo "FREEMEMORY=$free"
```

Executing it should give use memory values:

```
$ ./memory_probe
TOTALMEMORY=1020696
USEDMEMORY=209932
FREEMEMORY=810724
```

For real examples check the directories at `/var/lib/one/remotes/im`.

2.7.4 VM Information

The scripts should also provide information about the VMs running in the host. This is useful as it will only need one call to gather all that information about the VMs in each host. The output should be in this form:

```
VM_POLL=YES
VM=[
  ID=115,
  DEPLOY_ID=one-115,
```

(continues on next page)

(continued from previous page)

```
  VM_NAME=one-115,
  IMPORT_TEMPLATE="TkFNRT1vcGVubmVidWxhcm9ja3NiaWd0aW1lDQpDUFU9MQ0KTUVNT1JZPTIwMTQ4",
  POLL="STATE=a CPU=100.0 MEMORY=73232 NETRX=1518 NETTX=0 DISK_SIZE=[ ID=0, SIZE=24 ]␣
↪DISK_SIZE=[ ID=1, SIZE=0 ]  SNAPSHOT_SIZE=[ ID=1, DISK_ID=0, SIZE=24 ] SNAPSHOT_
↪SIZE=[ ID=0, DISK_ID=0, SIZE=24 ] " ]
VM=[
  ID=116,
  DEPLOY_ID=one-116,
  VM_NAME=one-115,
  IMPORT_TEMPLATE=
↪"TkFNRT1vcGVubmVidWxhcm9ja3NiaWd0aW11eWVzDQpDUFU9MQ0KTUVNT1JZPTIwNDg=",
  POLL="STATE=a CPU=100.5 MEMORY=77824 NETRX=1392 NETTX=0 DISK_SIZE=[ ID=0, SIZE=24 ]␣
↪DISK_SIZE=[ ID=1, SIZE=0 ]  " ]
VM=[
  ID=-1,
  DEPLOY_ID=f81d4fae-7dec-11d0-a765-00a0c91e6bf6,
  VM_NAME=MyVM,
  IMPORT_TEMPLATE="TkFNRT13aWxkdm0NCkNQVT0yDQpNRU1PUlk9MTAyNA==",
  POLL="STATE=a CPU=100.5 MEMORY=77824 NETRX=1392 NETTX=0 DISK_SIZE=[ ID=0, SIZE=24 ]␣
↪DISK_SIZE=[ ID=1, SIZE=0 ]  " ]
```

The first line (`VM_POLL=YES`) is used to indicate OpenNebula that VM information will follow. Then the information about the VMs is output in that form.

Key	Description
ID	OpenNebula VM id. It can be -1 in case this VM was not created by OpenNebula, a wild VM, that can be imported
DEPLOY_ID	Hypervisor name or identifier of the VM
VM_NAME	Human readable VM name (to show on import dialogs)
IMPORT_TEMPLATE	Base64 representation of the VM template to be used on import
POLL	VM monitoring info, in the same format as *VMM driver* poll

For example here is a simple script to get qemu/kvm VMs status from libvirt. As before, check the scripts from OpenNebula for a complete example:

```
#!/bin/bash

echo "VM_POLL=YES"

virsh -c qemu:///system list | grep one- | while read vm; do
    deploy_id=$(echo $vm | cut -d' ' -f 2)
    id=$(echo $deploy_id | cut -d- -f 2)
    status_str=$(echo $vm | cut -d' ' -f 3)

    if [ $status_str == "running" ]; then
        state="a"
    else
        state="e"
    fi

    echo "VM=["
    echo "  ID=$id,"
    echo "  DEPLOY_ID=$deploy_id,"
    echo "  POLL=\"STATE=$state\" ]"
```

(continues on next page)

(continued from previous page)

```
done
```

```
$ ./vm_poll
VM_POLL=YES
VM=[
  ID=0,
  DEPLOY_ID=one-0,
  POLL="STATE=a" ]
VM=[
  ID=1,
  DEPLOY_ID=one-1,
  POLL="STATE=a" ]
```

2.7.5 System Datastore Information

Information Manager drivers are also responsible to collect the datastore sizes and its available space. To do so there is an already made script that collects this information for filesystem and lvm based datastores. You can copy it from the KVM driver (`/var/lib/one/remotes/im/kvm-probes.d/monitor_ds.sh`) into your driver directory.

In case you want to create your own datastore monitor you have to return something like this in STDOUT:

```
DS_LOCATION_USED_MB=1
DS_LOCATION_TOTAL_MB=12639
DS_LOCATION_FREE_MB=10459
DS = [
  ID = 0,
  USED_MB = 1,
  TOTAL_MB = 12639,
  FREE_MB = 10459
]
DS = [
  ID = 1,
  USED_MB = 1,
  TOTAL_MB = 12639,
  FREE_MB = 10459
]
DS = [
  ID = 2,
  USED_MB = 1,
  TOTAL_MB = 12639,
  FREE_MB = 10459
]
```

These are the meanings of the values:

Variable	Description
DS_LOCATION_USED_MB	Used space in megabytes in the DATASTORE LOCATION
DS_LOCATION_TOTAL_MB	Total space in megabytes in the DATASTORE LOCATION
DS_LOCATION_FREE_MB	FREE space in megabytes in the DATASTORE LOCATION
ID	ID of the datastore, this is the same as the name of the directory
USED_MB	Used space in megabytes for that datastore
TOTAL_MB	Total space in megabytes for that datastore
FREE_MB	Free space in megabytes for that datastore

The DATASTORE LOCATION is the path where the datastores are mounted. By default is `/var/lib/one/datastores` but it is specified in the second parameter of the script call.

2.7.6 Creating a New IM Driver

Choosing the Execution Engine

OpenNebula provides two IM probe execution engines: `one_im_sh` and `one_im_ssh`. `one_im_sh` is used to execute probes in the frontend, for example `vcenter` uses this engine as it collects data via an API call executed in the frontend. On the other hand, `one_im_ssh` is used when probes need to be run remotely in the hosts, which is the case for `KVM`.

Populating the Probes

Both `one_im_sh` and `one_im_ssh` require an argument which indicates the directory that contains the probes. This argument is appended with ".d".

Making Use of collectd

If the new IM driver wishes to use the `collectd` component, it needs to:

- Use `one_im_ssh`
- The `/var/lib/one/remotes/im/<im_name>.d` should **only** contain 2 files, the sames that are provided by default inside `kvm.d`, which are: `collectd-client_control.sh` and `collectd-client.rb`.
- The probes should be actually placed in the `/var/lib/one/remotes/im/<im_name>-probes.d` folder.

Enabling the Driver

A new IM section should be placed added to `oned.conf`.

Example:

```
IM_MAD = [
      name       = "ganglia",
      executable = "one_im_sh",
      arguments  = "ganglia" ]
```

2.8 Networking Driver

This component is in charge of configuring the network in the hypervisors. The purpose of this guide is to describe how to create a new network manager driver.

2.8.1 Driver Configuration and Description

To enable a new network manager driver, the frist requirement is to make a new directory with the name of the driver in `/var/lib/one/remotes/vnm/remotes/<name>` with three files:

- **Pre**: This driver should perform all the network related actions required before the Virtual Machine starts in a host.
- **Post**: This driver should perform all the network related actions required after the Virtual Machine starts (actions which typically require the knowledge of the `tap` interface the Virtual Machine is connected to).
- **Clean**: If any clean-up should be performed after the Virtual Machine shuts down, it should be placed here.

Warning: The above three files **must exist**. If no action is required in them a simple `exit 0` will be enough.

Warning: Remember that any change in the `/var/lib/one/remotes` directory won't be effective in the Hosts until you execute, as oneadmin: `onehost sync -f`

Virtual Machine actions and their relation with Network actions:

- **Deploy**: `pre` and `post`
- **Shutdown**: `clean`
- **Cancel**: `clean`
- **Save**: `clean`
- **Restore**: `pre` and `post`
- **Migrate**: `pre` (target host), `clean` (source host), `post` (target host)
- **Attach Nic**: `pre` and `post`
- **Detach Nic**: `clean`

After that you need to define the bridging technology used by the driver at `/etc/one/oned.conf`. OpenNebula support three differents technologies, **Linux Bridge**, **Open vSwitch** and **vCenter Port Groups**. See the examples below:

```
VN_MAD_CONF = [
    NAME = "vcenter"
    BRIDGE_TYPE = "vcenter_port_groups"
]

VN_MAD_CONF = [
    NAME = "ovswitch_vxlan"
    BRIDGE_TYPE = "openvswitch"
]

VN_MAD_CONF = [
    NAME = "bridge"
    BRIDGE_TYPE = "linux"
]

VN_MAD_CONF = [
    NAME = "custom"
    BRIDGE_TYPE = "none"
]
```

Note: Note that you can set BRIDGE_TYPE attribute to `none` if you need to leave the bridge empty.

2.8.2 Driver Customization

Default driver actions support the execution of hooks after the main action is succesfully executed. In order to create an action hook you need to place your custom configuration scripts in the corresponding **action.d** directory, inside the target networking driver directory. Files found in that directory will be run in an alphabetical order, excluding the ones oneadmin cannot run due to lack of permissions. If the main action fails the hooks won't be run. If a hook fails the corresponding network actions will be consider as a **FAILURE** and the VM will change its state accordingly. Note that the scripts will receive the same information as the main action through stdin.

For example, this is the directory tree of the bridge driver synced to a virtualization node with some custom scripts

```
root@ubuntu1804-lxd-ssh-6ee11-2:/var/tmp/one/vnm/bridge# tree ./
./
├── clean
├── clean.d
│   ├── 01_del_fdb
│   └── 02_del_routes
├── post
├── post.d
│   ├── 01_add_fdb
│   └── 02_add_routes
├── pre
├── pre.d
│   └── 01_update_router
└── update_sg
```

2.8.3 Driver Parameters

All three driver actions receive the **XML VM template** encoded in base64 format by `stdin` and a parameter which is the **deploy-id** of the Virtual Machine e.g.: `one-17`.

The `clean` action doesn't require **deploy-id**

2.8.4 The 802.1Q Driver

Driver Files

The code can be enhanced and modified, by chaning the following files in the frontend:

- /var/lib/one/remotes/vnm/802.1Q/post
- /var/lib/one/remotes/vnm/802.1Q/vlan_tag_driver.rb
- /var/lib/one/remotes/vnm/802.1Q/clean
- /var/lib/one/remotes/vnm/802.1Q/pre

Driver Actions

Action	Description
Pre	Creates a VLAN tagged interface in the Host and a attaches it to a dynamically created bridge.
Post	N/A
Clean	It doesn't do anything. The VLAN tagged interface and bridge are kept in the Host to speed up future VMs

2.8.5 The VXLAN Driver

Driver Files

The code can be enhanced and modified, by changing the following files in the frontend:

- /var/lib/one/remotes/vnm/vxlan/vxlan_driver.rb
- /var/lib/one/remotes/vnm/vxlan/post
- /var/lib/one/remotes/vnm/vxlan/clean
- /var/lib/one/remotes/vnm/vxlan/pre

Driver Actions

Action	Description
Pre	Creates a VXLAN interface through PHYDEV, creates a bridge (if needed) and attaches the vxlan device.
Post	When the VM is associated to a security group, the corresponding iptables rules are applied.
Clean	It doesn't do anything. The VXLAN interface and bridge are kept in the Host to speed up future VMs

2.8.6 The Open vSwitch Driver

Driver Actions

Action	Description
Pre	N/A
Post	Performs the appropriate Open vSwitch commands to tag the virtual tap interface.
Clean	It doesn't do anything. The virtual tap interfaces will be automatically discarded when the VM is shut down.

2.8.7 The ebtables Driver

Driver Actions

Action	Description
Pre	N/A
Post	Creates EBTABLES rules in the Host where the VM has been placed.
Clean	Removes the EBTABLES rules created during the `Post` action.

2.9 Authentication Driver

This guide will show you how to develop a new driver for OpenNebula to interact with an external authentication service.

OpenNebula comes with an internal user/password way of authentication, this is called `core`. To be able to use other auth methods there is a system that performs authentication with external systems. Authentication drivers are responsible of getting the user credentials from OpenNebula database and login and answer whether the authentication is correct or not.

In the OpenNebula database there are two values saved for every user, this is `username` and `password`. When the driver used for authentication is core (authenticated without an external auth driver) the password value holds the SHA1 hash of the user's password. In case we are using other authentication method this `password` field can contain any other information we can use to recognize a user, for example, for x509 authentication this field contains the user's public key.

2.9.1 Authentication Driver

Authentication drivers are located at `/var/lib/one/remotes/auth`. There is a directory for each of authentication drivers with an executable inside called `authenticate`. The name of the directory has to be the same as the user's auth driver we want to authenticate. For example, if a user has as auth driver `x509` OpenNebula will execute the file `/var/lib/one/remotes/auth/x509/authenticate` when he performs an OpenNebula action.

The authentication driver expects parameters passed on the standard input as the XML document with following structure:

```
<AUTHN>
    <USERNAME>VALUE</USERNAME>
    <PASSWORD>VALUE</PASSWORD>
    <SECRET>VALUE</SECRET>
</AUTHN>
```

Where:

- `USERNAME`: name of the user who wants to authenticate.
- `PASSWORD`: value of the password field for the user that is trying to authenticate. This can be – when the user does not exist in the OpenNebula database.
- `SECRET`: value provided in the password field of the authentication string.

Warning: Before the OpenNebula 5.6, the parameters were passed as command line parameters. Now all the data are passed only on standard input only!

For example, we can create a new authentication method that just checks the length of the password. For this we can store in the password field the number of characters accepted, for example 5, and user name test. Here are some example calls to the driver with several passwords:

```
echo '<AUTHN><USERNAME>test</USERNAME><PASSWORD>5</PASSWORD><SECRET>testpassword</
↪SECRET></AUTHN>' | \
    authenticate

echo '<AUTHN><USERNAME>test</USERNAME><PASSWORD>5</PASSWORD><SECRET>another_try</
↪SECRET></AUTHN>' | \
    authenticate

echo '<AUTHN><USERNAME>test</USERNAME><PASSWORD>5</PASSWORD><SECRET>12345</SECRET></
↪AUTHN>' | \
    authenticate
```

The script should exit with a non 0 status when the authentication is not correct and write in `stderr` the error. When the authentication is correct it should return:

- Name of the driver. This is used when the user does not exist, this will be written to user's the auth driver field.
- User name
- Text to write in the user's password field in case the user does not exist.

The code for the `/var/lib/one/remotes/auth/length/authenticate` executable can be:

```
#!/bin/bash

data=$(cat -)

username=$(echo "${data}" | xmllint --xpath '//AUTHN/USERNAME/text()' -)
password=$(echo "${data}" | xmllint --xpath '//AUTHN/PASSWORD/text()' -)
secret=$(echo "${data}" | xmllint --xpath '//AUTHN/SECRET/text()' -)

length=$(echo -n "$secret" | wc -c | tr -d ' ')

if [ $length = $password ]; then
    echo "length $username $secret"
else
    echo "Invalid password"
    exit 255
fi
```

2.9.2 Enabling the Driver

To be able to use the new driver we need to add its name to the list of enabled drivers in `oned.conf`:

```
AUTH_MAD = [
    executable = "one_auth_mad",
    authn = "ssh,x509,ldap,server_cipher,server_x509,length"
]
```

2.10 Cloud Bursting Driver

This guide will show you how to develop a new driver for OpenNebula to interact with an external cloud provider.

Cloud bursting, or hybrid cloud, is a model in which the local resources of a Private Cloud are combined with resources from remote Cloud providers. The remote provider could be a commercial Cloud service, such as Amazon EC2 or Microsoft Azure, or a partner infrastructure running a different OpenNebula instance. Such support for cloud bursting enables highly scalable hosting environments. For more information on this model see the Cloud Bursting overview

The remote cloud provider will be included in the OpenNebula host pool like any other physical host of your infrastructure:

```
$ onehost create remote_provider im_provider vmm_provider

$ onehost list
  ID NAME            CLUSTER   RVM      ALLOCATED_CPU      ALLOCATED_MEM STAT
   2 kvm-            -           0       0 / 800 (0%)      0K / 16G (0%) on
   3 kvm-1           -           0       0 / 100 (0%)     0K / 1.8G (0%) on
   4 remote_provider -           0       0 / 500 (0%)     0K / 8.5G (0%) on
```

When you create a new host in OpenNebula you have to specify the following parameters:

- Name: remote_provider

Name of the host, in case of physical hosts it will the ip address or hostname of the host. In case of remote providers it will be just a name to identify the provider.

- *Information Manager*: `im_provider`

IM driver gather information about the physical host and hypervisor status, so the OpenNebula scheduler knows the available resources and can deploy the virtual machines accordingly.

- *VirtualMachine Manager*: `vmm_provider`

VMM drivers translate the high-level OpenNebula virtual machine life-cycle management actions, like deploy, shutdown, etc. into specific hypervisor operations. For instance, the KVM driver will issue a virsh create command in the physical host. The EC2 driver translate the actions into Amazon EC2 API calls.

Note: Storage and Network drivers are derived from the VM characteristics rather than the host.

When creating a new host to interact with a remote cloud provider we will use mock versions for the TM and VNM drivers. Therefore, we will only implement the functionality required for the IM and VMM driver.

2.10.1 Adding the Information Manager

Edit oned.conf

Add a new IM section for the new driver in oned.conf:

```
#*******************************************************************************
# Information Driver Configuration
#*******************************************************************************
# You can add more information managers with different configurations but make
# sure it has different names.
#
#   name      : name for this information manager
#
#   executable: path of the information driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable, usually a probe configuration file,
#               can be an absolute path or relative to $ONE_LOCATION/etc (or
#               /etc/one/ if OpenNebula was installed in /)
#    -r number of retries when monitoring a host
#    -t number of threads, i.e. number of hosts monitored at the same time
#*******************************************************************************
#-------------------------------------------------------------------------------
#  EC2 Information Driver Manager Configuration
#-------------------------------------------------------------------------------
IM_MAD = [
      name       = "im_provider",
      executable = "one_im_sh",
      arguments  = "-t 1 -r 0 provider_name" ]
#-------------------------------------------------------------------------------
```

Populating the Probes

Create a new directory to store your probes, the name of this folder must match the name provided in the arguments section of the IM_MAD in oned.conf:

- /var/lib/one/remotes/im/<provider_name>.d

These probes must return:

- *Information of the host capacity*, to limit the number of VMs that can be deployed in this host.
- *Information of the VMs* running in this host.

You can see an example of these probes in the ec2 driver (code) included in OpenNebula

You must include the PUBLIC_CLOUD and HYPERVISOR attributes as one of the values returned by your probes, otherwise OpenNebula will consider this host as local. The HYPERVISOR attribute will be used by the scheduler and should match the TYPE value inside the PUBLIC_CLOUD section provided in the VM template.

```
PUBLIC_CLOUD="YES"
HYPERVISOR="provider_name"
```

2.10.2 Adding the Virtual Machine Manager

Edit oned.conf

```
#*******************************************************************************
# Virtualization Driver Configuration
#*******************************************************************************
# You can add more virtualization managers with different configurations but
# make sure it has different names.
#
#   name      : name of the virtual machine manager driver
#
#   executable: path of the virtualization driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable
#    -r number of retries when monitoring a host
#    -t number of threads, i.e. number of hosts monitored at the same time
#
#   default   : default values and configuration parameters for the driver, can
#               be an absolute path or relative to $ONE_LOCATION/etc (or
#               /etc/one/ if OpenNebula was installed in /)
#
#   type      : driver type, supported drivers: xen, kvm, xml
#
#-------------------------------------------------------------------------------
VM_MAD = [
    name       = "vmm_provider",
    executable = "one_vmm_sh",
    arguments  = "-t 15 -r 0 provider_name",
    type       = "xml" ]
#-------------------------------------------------------------------------------
```

Create the Driver Folder and Implement the Specific Actions

Create a new folder inside the remotes dir (/var/lib/one/remotes/vmm). The new folder should be named "provider_name", the name specified in the previous VM_MAD arguments section.

This folder must contain scripts for the supported actions. You can see the list of available actions in the *Virtual Machine Driver guide*. These scripts are language-agnostic so you can implement them using python, ruby, bash...

You can see examples on how to implement this in the ec2 driver:

- EC2 Shutdown action:

```
#!/usr/bin/env ruby

# ---------------------------------------------------------------------------- #
# Copyright 2002-2016, OpenNebula Project, OpenNebula Systems                  #
#                                                                              #
# Licensed under the Apache License, Version 2.0 (the "License"); you may     #
# not use this file except in compliance with the License. You may obtain      #
# a copy of the License at                                                     #
#                                                                              #
# http://www.apache.org/licenses/LICENSE-2.0                                   #
#                                                                              #
# Unless required by applicable law or agreed to in writing, software          #
# distributed under the License is distributed on an "AS IS" BASIS,            #
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.     #
# See the License for the specific language governing permissions and          #
# limitations under the License.                                               #
# ---------------------------------------------------------------------------- #

$: << File.dirname(__FILE__)

require 'ec2_driver'

deploy_id = ARGV[0]
host      = ARGV[1]

ec2_drv = EC2Driver.new(host)

ec2_drv.shutdown(deploy_id)
```

Create the New Host

After restarting oned we can create the new host that will use this new driver

```
$ onehost create remote_provider im_provider vmm_provider
```

Create a new Virtual Machine

Create a new VM using a template with an specific section for this provider. You have to include the required information to start a new VM inside the PUBLIC_CLOUD section, and the TYPE attribute must match the HYPERVISOR value of the host. For example:

```
$ cat vm_template.one
CPU=1
```

(continues on next page)

(continued from previous page)

```
MEMORY=256
PUBLIC_CLOUD=[
    TYPE=provider_name
    PROVIDER_IMAGE_ID=id-141234,
    PROVIDER_INSTANCE_TYPE=small_256mb
]

$ onevm create vm_template
ID: 23

$ onevm deploy 23 remote_provider
```

After this, the deploy script will receive the following arguments:

- The path to the deployment file that contains the following XML:

```
<CPU>1</CPU>
<MEMORY>256</MEMORY>
<PUBLIC_CLOUD>
    <TYPE>provider_name</TYPE>
    <PROVIDER_IMAGE_ID>id-141234</PROVIDER_IMAGE_ID>
    <PROVIDER_INSTANCE_TYPE>small_256mb</PROVIDER_INSTANCE_TYPE>
</PUBLIC_CLOUD>
```

- The hostname: `remote_provider`
- The VM ID: `23`

The deploy script has to return the ID of the new resource and an exit_code 0:

```
$ cat /var/lib/one/remote/provider/deploy
#!/bin/bash
deployment_file=$1
# Parse required parameters from the template
..
# Retrieve account credentials from a local file/env
...
# Create a new resource using the API provider
...
# Return the provider ID of the new resource and exit code 0 or an error message
```

2.11 Market Driver

The Market Driver is in charge of managing both MarketPlaces and MarketPlaceApps.

2.11.1 MarketPlace Drivers Structure

The main drivers are located under `/var/lib/one/remotes/market/<market_mad>`.

import

The `import` action is a special action that involves two driver calls chained one after the other. The involved actors are: `/var/lib/one/remotes/datastore/<ds_mad>/export` and `/var/lib/one/remotes/`

market/<market_mad>/import. Note they they aren't piped together: the core will run the export action first, collect the output and use it for the driver message of the import action.

<ds_mad>/export:

The job of the export is to:

- Calculate the MD5, FORMAT, SIZE
- Generate an IMPORT_SOURCE so the <market_mad>/import can do the image => market dump
- Specify DISPOSE="YES" if the IMPORT_SOURCE is a temporary file that must be removed after the dump performed by <market_mad>/import. DISPOSE="NO" if otherwise.

ARGUMENTS: driver_dump_xml image_id. The driver_dump_xml is an XML dump of the driver action encoded in Base 64, which contains all the required information: image_data and ds_data.

RETURNS

It should return an XML document:

```
<IMPORT_INFO>
    <IMPORT_SOURCE>$IMPORT_SOURCE</IMPORT_SOURCE>
    <MD5>$MD5_SUM</MD5>
    <SIZE>$SIZE</SIZE>
    <FORMAT>$FORMAT</FORMAT>
    <DISPOSE>NO</DISPOSE>
</IMPORT_INFO>"
```

<market_mad>/import:

The job of the export is to grab the IMPORT_SOURCE and dump it to the backend.

ARGUMENTS: driver_dump_xml image_id. The driver_dump_xml is an XML dump of the driver action encoded in Base 64, which contains all the required information: app_data, market_data, image_data and ds_data.

RETURNS

It should return this OpenNebula-syntax template:

```
SOURCE="<SOURCE>"
MD5="<MD5>"
SIZE="<SIZE_IN_MB>"
FORMAT="<FORMAT>"
```

Note that typically inside the import script we will find a call to the downloader.sh as such:

```
${UTILS_PATH}/downloader.sh $IMPORT_SOURCE -
```

Which will be in turn piped to the target destination in the MarketPlace, or in some Market Drivers the target file will appear in the downloader.sh command as the destination instead of -. Note that this might require extending downloader.sh to handle a custom new protocol, like: s3://, rbd://, etc...

monitor

The monitor action is orchestrated by the /var/lib/one/remotes/market/<market_mad>/monitor script.

There are two kinds of monitor scripts, those that do not require the discovery of the MarketPlaceApps, since OpenNebula already know about them because it has created them. And those that include this discovery, like in the case of the one/monitor script to monitor the official OpenNebula Systems MarketPlace.

ARGUMENTS: `driver_dump_xml market_id`. The `driver_dump_xml` is an XML dump of the driver action encoded in Base 64, which contains all the required information: `market_data`.

RETURNS

In the case of simple monitoring with no discovery, the `monitor` action must return a report of the available and used space in the MarketPlace in OpenNebula-syntax template format:

```
USED_MD=<USED_MB>
FREE_MB=<FREE_MB>
TOTAL_MB=<TOTAL_MB>
```

In the case of monitoring with discovery, it must also return the information of each Appliance, encoded in Base64 in the following way (the full APP string has been trimmed for legibility):

```
APP="TkFNRT0idHR5b..."
APP="TkFNRT0idHR5b..."
APP="TkFNRT0iQ2Fya..."
APP="TkFNRT0iVGVzd..."
APP="TkFNRT0iZ1VzZ..."
APP="TkFNRT0iVnlhd..."
APP="TkFNRT0iZ1VTR..."
APP="TkFNRT0iZGVia..."
...
```

If we unpack one of these APPs we will obtain the following:

```
NAME="ttylinux - kvm"
SOURCE="http://marketplace.opennebula.systems//appliance/4fc76a938fb81d3517000003/
↪download/0"
IMPORT_ID="4fc76a938fb81d3517000003"
ORIGIN_ID="-1"
TYPE="IMAGE"
PUBLISHER="OpenNebula.org"
FORMAT="raw"
DESCRIPTION="This is a very small image that works with OpenNebula. It's already␣
↪contextualized. The purpose of this image is to test OpenNebula deployments,␣
↪without wasting network bandwith thanks to the tiny footprint of this image
(40MB)."
VERSION="1.0"
TAGS="linux, ttylinux,  4.8,  4.10"
SIZE="40"
MD5="04c7d00e88fa66d9aaa34d9cf8ad6aaa"
VMTEMPLATE64=
↪"Q09OVEVYVCA9IFsgTkVUV09SSyAgPSJZRVMiLFNTSF9QVUJMSUNfS0VZICA9IiRVU0VSW1NTSF9QVUJMSUNfS0VZXSJdCgpDUI
↪"
```

Which is the MarketPlaceApp template in OpenNebula-syntax format.

export

The `export` job is again two-fold:

- Create a new image by calling `<ds_mad>/cp`
- Create a new template, if it exists in the MarketPlaceApp (`VMTEPLATE64`)

There is no specific `<market_mad>` driver file associated with this job, it actually calls an already existing driver, the `<ds_mad>/cp`. Please read the *Storage Driver* guide to learn more about this driver action.

It is worth noting that the MarketPlaceApp's `IMPORT_SOURCE` field will be used as the `PATH` argument for the `<ds_mad>/cp` action. Therefore, this action must understand that `IMPORT_SOURCE` which in turn means that `downloader.sh` must understand it too.

delete

This job deletes a MarketPlaceApp.

ARGUMENTS: `driver_dump_xml image_id`. The `driver_dump_xml` is an XML dump of the driver action encoded in Base 64, which contains all the required information: `market_data` and `marketapp_data`.

RETURNS: No return message.

2.12 IPAM driver

A IPAM driver lets you delegate IP lease management to an external component. This way you can coordinate IP use with other virtual or bare metal servers in your datacenter. To effectively use an external IPAM you need to develop four action scripts that hooks on different points of the IP network/lease life-cycle.

Note that OpenNebula includes a built-in internal IPAM. You need to develop this component if you are using a IPAM server and want to coordinate OpenNebula with it.

2.12.1 IPAM Drivers Structure

The main drivers are located under `/var/lib/one/remotes/ipam/<ipam_mad>`. For an example, you can take a look to `/var/lib/one/remotes/ipam/dummy`. This set of simple scripts can be used as an starting point to develop the integration.

register_address_range

This action is used to register a new IP network in the IPAM. The network may be selected from a pool of free networks or if an specific network is requested its availability maybe checked by the IPAM driver. The IPAM driver must return an OpenNebula AddressRange definition, potentially augmented with network specific variables to be used by VMs (e.g. `GATEWAY`, `NETWORK_MASK`...)

STDIN Argument:

- AddressRange. The AddressRange (IP network) request in XML encoded in Base 64. The XML may contain any of the attributes used to define an AR. Note that OpenNebula uses a free format for objects so you can freely add more and process more (or less) attributes in this action. At least TYPE, IPAM_MAD and SIZE will be present:

```
<IPAM_DRIVER_ACTION_DATA>
<AR>
  <TYPE>IP4</TYPE>
  <IP> First IP in the network in '.' notation </IP>
  <MAC> First MAC in the network in ':' notation </MAC
  <SIZE>Number of IPs in the network </SIZE>
  <NETWORK_ADDRESS> Base network address</NETWORK_ADDRESS>
  <NETWORK_MASK> Network mask</NETWORK_MASK>
  <GATEWAY> Default gateway for the network</GATEWAY>
  <DNS> DNS servers, a space separated list of servers</DNS>
  <GUEST_MTU> Sets the MTU for the NICs in this network</GUEST_MTU>
```

(continues on next page)

(continued from previous page)

```
  <SEARCH_DOMAIN> for DNS client</SEARCH_DOMAIN>
</AR>
</IPAM_DRIVER_ACTION_DATA>
```

Arguments:

- Request ID, used internally to identify this IPAM request.

Returns: It should return throudh STDOUT a valid AddressRange definiton in template format. The response MUST include IPAM_MAD, TYPE, IP and SIZE attribute. For example:

- A basic network definition

```
AR = [
  IPAM_MAD = "dummy",
  TYPE = "IP4",
  IP   = "10.0.0.1",
  SIZE = "255"
]
```

- A complete network definition. Custom attributes (free form, key-value) can be added, names cannot be repeated.

```
AR = [
  IPAM_MAD = "dummy",
  TYPE = "IP4",
  IP   = "10.0.0.2",
  SIZE = "200",
  NETWORK_ADDRESS   = "10.0.0.0",
  NETWORK_MASK      = "255.255.255.0",
  GATEWAY           = "10.0.0.1",
  DNS               = "10.0.0.1",
  IPAM_ATTR         = "10.0.0.240",
  OTHER_IPAM_ATTR   = ".mydoamin.com"
]
```

unregister_address_range

This action is used to unregister an address range from the IPAM.

STDIN Argument:

- AddressRange. The AddressRange in XML encoded in Base 64. The XML may contain any of the attributes used to define an AR. Note that OpenNebula uses a free format for objects so you can freely add more and process more (or less) attributes in this action. At least TYPE, IPAM_MAD and SIZE will be present:

```
<IPAM_DRIVER_ACTION_DATA>
<AR>
  <TYPE>IP4</TYPE>
  <IP> First IP in the network in '.' notation </IP>
  <MAC> First MAC in the network in ':' notation </MAC
  <SIZE>Number of IPs in the network </SIZE>
  <NETWORK_ADDRESS> Base network address</NETWORK_ADDRESS>
  <NETWORK_MASK> Network mask</NETWORK_MASK>
  <GATEWAY> Default gateway for the network</GATEWAY>
  <DNS> DNS servers, a space separated list of servers</DNS>
```

(continues on next page)

(continued from previous page)

```
  <GUEST_MTU> Sets the MTU for the NICs in this network</GUEST_MTU>
  <SEARCH_DOMAIN> for DNS client</SEARCH_DOMAIN>
</AR>
</IPAM_DRIVER_ACTION_DATA>
```

Arguments:

- Request ID, used internally to identify this IPAM request.

Returns: This scripts MUST exit 0 if no errors we found.

allocate_address

This script is used to register an specific IP address (or addresses) as used. The IP (or IPs) will be used by an OpenNebula VM and should not be allocated to any other host in the network.

STDIN Argument:

- AddressRange and Address. The AddressRange (IP network) and address request in XML, encoded in Base 64. The XML will contain the AR as defined by the previous action; and the address request:

```
<IPAM_DRIVER_ACTION_DATA>
<AR>
  As returned by previous action, see above for examples.
</AR>
<ADDRESS>
  <IP> Requested IP address </IP>
  <SIZE> Number of IPs to allocate, in a continous range from the firs IP</SIZE>
  <MAC> Optional the MAC address </MAC>
</ADDRESS>
</IPAM_DRIVER_ACTION_DATA>
```

Arguments:

- Request ID, used internally to identify this IPAM request.

Returns: This scripts MUST exit 0 if the address is free.

get_address

This script is used to lease an IP address (or addresses). The IP (or IPs) will be used by an OpenNebula VM and should not be allocated to any other host in the network.

STDIN Argument:

- AddressRange and Address. The AddressRange (IP network) and address request in XML, encoded in Base 64. The XML will contain the AR as defined by the previous action; and the address request:

```
<IPAM_DRIVER_ACTION_DATA>
<AR>
  As returned by previous action, see above for examples.
</AR>
<ADDRESS>
  <SIZE> Number of IPs to allocate, in a continous range from the firs IP</SIZE>
</ADDRESS>
</IPAM_DRIVER_ACTION_DATA>
```

Arguments:

- Request ID, used internally to identify this IPAM request.

Returns: This script MUST output the leased IP range as defined by the ADDRESS element in template format thourgh STOUT. For example, to lease IPs from 10.0.0.2 to 10.0.0.35 return:

```
ADDRESS = [ IP = "10.0.0.2", SIZE=34 ]
```

If the "size" IPs cannot be assgined the sript must return -1, otherwise it must exit 0.

free_address

This script is used to free an specific IP address (or addresses). The IP (or IPs) are no longer in use by OpenNebula VMs or reservations.

STDIN Argument:

- AddressRange and Address. Same as in `allocate_address`.

Arguments:

- Request ID, used internally to identify this IPAM request.

Returns: This scripts MUST exit 0 if the address is free.

2.12.2 IPAM Usage

To use your new IPAM module you need to:

- Place the four previous scripts in `/var/lib/one/remotes/ipam/<ipam_mad>`.
- Activate the driver in oned.conf by adding the IPAM driver name (same as the one used to name the directory where the action scripts are stored) to the `-i` option of the IPAM_MAD attribure:

```
IPAM_MAD = [
    EXECUTABLE = "one_ipam",
    ARGUMENTS  = "-t 1 -i dummy, <ipam_mad>"
]
```

- You need to restart OpenNebula to load the new ipam module.
- Now, define Virtual Networks to use the IPAM. Add `IPAM_MAD` to the AR defintion, for example:

```
NAME = "IPAM Network"

BRIDGE  = "br0"
VNM_MAD = "dummy"

AR = [
  SIZE     = 21,
  IPAM_MAD = <ipam_mad>
 ]
```

Any VM (or VNET reservation) from IPAM Network will contanct the IPAM using your drivers.

2.13 vCenter Driver

The vCenter driver for OpenNebula enables the interaction with vCenter to control the life-cycle of vCenter resources such as Virtual Machines, Templates, Networks and VMDKs.

2.13.1 OpenNebula approach to vCenter interaction

OpenNebula consumes resources from vCenter using import tools, although OpenNebula can create on its own switches, port groups and VMDK files. OpenNebula uses vCenter object references in order to control resources.

Managed Object Reference

vCenter resources like VMs, Templates, Datastores, Networks, Hosts and many more have an object identifier called Managed Object Reference, or moref, which is used by vCenter and OpenNebula to locate and manage these resources. Morefs have the following pattern:

- `vm-XXXXX` for Virtual Machines and Templates e.g `vm-2543`
- `datastore-XXXXX` for Datastores e.g `datastore-411`
- `network-XXXXX` for vCenter port groups e.g `network-17`
- `domain-XXXXX` for vCenter clusters e.g `domain-c7`
- `group-pXXXXX` for Storage DRS Clusters e.g `group-p1149`

OpenNebula stores the moref in attributes inside templates definitions. These are the attributes used by OpenNebula:

- `VCENTER_CCR_REF`. Contains a cluster moref.
- `VCENTER_NET_REF`. Contains a port group moref.
- `VCENTER_DS_REF`. Contains a datastore moref.
- `VCENTER_DC_REF`. Contains a datacenter moref.
- `VCENTER_TEMPLATE_REF`. Contains a VM template moref.
- `DEPLOY_ID`. When a VM is instantiated or imported, it will contain the VM's moref.

The Managed Object Reference is a unique identifier in a specific vCenter instance, so we could find two resources with the same moref in two different vCenter servers. That's why a vCenter Instance UUID is used together with the moref to uniquely identify a resource. An instance uuid has a pattern like this 805af9ee-2267-4f8a-91f5-67a98051eebc.

OpenNebula stores the instance uuid inside a template definition in the following attribute:

- `VCENTER_INSTANCE_ID`

OpenNebula's import tools, which are explained later, will get the morefs and vcenter's intance uuid for you when a resource is imported or created, but if you want to know the managed object reference of a resource we offer you the following information.

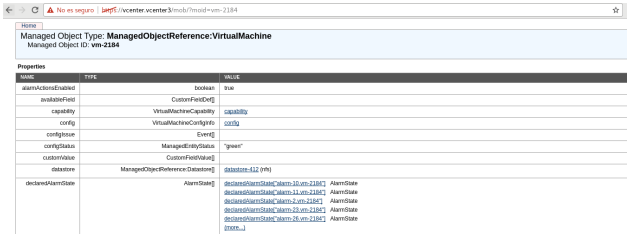
Getting a Managed Object Reference

In vSphere's web client, when we click on a resource in the Inventory like a Virtual Machine, the URL in your browser's changes. If you examine the URL at the end, you'll find a parameter ServerObjectRef which is followed by the instance uuid and you'll also find the resource type (VirtualMachine, Datastore...) followed by the moref.

Here you have two examples where you can find the instance uuid and the moref:

If you want to get information about a resource, you can use the Managed Object Browser provided by VMWare, which is listens on https://x.x.x.x/mob/ where x.x.x.x is the FQDN/IP of your vCenter instance. Use your vSphere credentials and you can browse a resource using an URL like this https://x.x.x.x/mob/?moid=yyyyy where yyyyy is the moref of the resource you want to browse.

This a screenshot of a virtual machine browsed by the Managed Object Browser:



VMware VM Templates and OpenNebula

In OpenNebula, Virtual Machines are deployed from VMware VM Templates that **must exist previously in vCenter and must be imported into OpenNebula**. There is a one-to-one relationship between each VMware VM Template and the equivalent OpenNebula VM Template. Users will then instantiate the OpenNebula VM Template and OpenNebula will create a Virtual Machine clone from the vCenter template.

Note: VMs launched in vCenter through OpenNebula can access VMware advanced features and it can be managed through the OpenNebula provisioning portal -to control the life-cycle, add/remove NICs, make snapshots- or through vCenter (e.g. to move the VM to another datastore or migrate it to another ESX). OpenNebula will poll vCenter to detect these changes and update its internal representation accordingly.

Note: Changes to the VM Template in vCenter (like for instance migrate its disk to another datastore) are not supported. If any change is made to the vCenter VM Template, it needs to be reimported into OpenNebula.

There is no need to convert your current Virtual Machines or Templates, or import/export them through any process; once ready just save them as VM Templates in vCenter, following this procedure and import it into OpenNebula as explained later in this chapter.

When a VMware VM Template is imported, OpenNebula will detect any virtual disk and network interface within the template. For each virtual disk, OpenNebula will create an OpenNebula image representing each disk discovered in the template. In the same way, OpenNebula will create a network representation for each standard or distributed port group associated to virtual network interfaces found in the template.

Warning: The process that discovers virtual disks and networks and then creates OpenNebula representations take some time depending on the number of discovered resources and the operations that must be performed so be patient.

The following sections explain some features that are related with vCenter templates and virtual machines deployed by OpenNebula.

Linked Clones

In OpenNebula, a new VM is deployed when a clone of an existing vCenter template is created, that's why OpenNebula requires that templates are first created in vCenter and then imported into OpenNebula.

In VMWare there are two types of cloning operations:

- The Full Clone. A full clone is an independent copy of a template that shares nothing with the parent template after the cloning operation. Ongoing operation of a full clone is entirely separate from the parent template. This is the default clone action in OpenNebula.
- The Linked Clone. A linked clone is a copy of a template that shares virtual disks with the parent template in an ongoing manner. This conserves disk space, and allows multiple virtual machines to use the same software installation.

When the **onevcenter** tool is used to import a vCenter template, as explained later, you'll be able to specify if you want to use linked clones when the template is imported. Note that if you want to use linked clones, OpenNebula has to create delta disks on top of the virtual disks that are attached to the template. This operation will modify the template so you may prefer that OpenNebula creates a copy of the template and modify that template instead, the onevcenter tool will allow you to choose what you prefer to do.

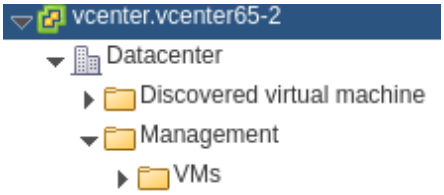
Important: Linked clone disks cannot be resized.

Note: Sunstone does not allow you to specify if you want to use Linked Clones as the operations involved are heavy enough to keep them out of the GUI.

VM Placement

In OpenNebula, by default, a new virtual machine cloned from a vCenter template will be displayed in the same folder where the template lives in vSphere's VM and Templates inventory. However you have the chance to select in which folder you want to see the VM's based on that template.

For example, if you have the following directory tree and you want VMs to be placed in the VMs folder under Management, the path to that folder from the datacenter root would be /Management/VMs. You can use that path in different OpenNebula actions e.g when a template is imported.



Resource Pools in OpenNebula

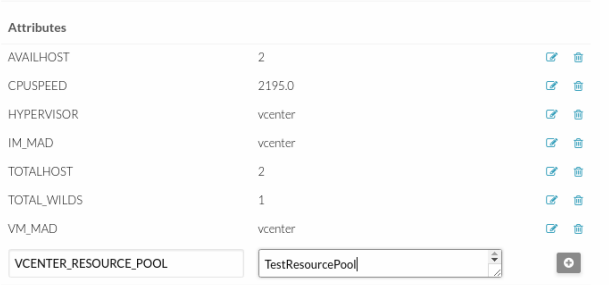
OpenNebula can place VMs in different Resource Pools. There are two approaches to achieve this:

- fixed per Cluster basis
- flexible per VM Template basis.

Fixed per Cluster basis

In the fixed per Cluster basis approach, the vCenter connection that OpenNebula use can be confined into a Resource Pool, to allow only a fraction of the vCenter infrastructure to be used by OpenNebula users. The steps to confine OpenNebula users into a Resource Pool are:

- Create a new vCenter user.
- Create a Resource Pool in vCenter and assign the subset of Datacenter hardware resources wanted to be exposed through OpenNebula.
- Give vCenter user Resource Pool Administration rights over the Resource Pool.
- Give vCenter user Resource Pool Administration (or equivalent) over the Datastores the VMs are going to be running on.
- Import the vCenter cluster into OpenNebula as explained later. The import action will create an OpenNebula host.
- Add a new attribute called VCENTER_RESOURCE_POOL to OpenNebula's host template representing the vCenter cluster (for instance, in the info tab of the host, or in the CLI), with the reference to a Resource Pool.



Flexible per VM Template

The second approach is more flexible in the sense that all Resource Pools defined in vCenter can be used, and the mechanism to select which one the VM is going to reside into can be defined using the attribute VCENTER_RESOURCE_POOL in the OpenNebula VM Template.

Once we have in OpenNebula an imported template, we can **update it** from the CLI or the Sunstone interface and we will have two choices:

- Specify a fixed Resource Pool that will be used by any VM based on the template.
- Offer a list of Resource Pools so the user can select one of them when a VM is instantiated.

Using the CLI we would use the **onetemplate update** command and we would add or edit the VCENTER_RESOURCE_POOL attribute.

If we want to specify a Resource Pool, that attribute would be placed inside the template and would contain a reference to the resource pool.

```
VCENTER_RESOURCE_POOL="TestResourcePool/NestedResourcePool"
```

If we wanted to offer a list to the user, we would place the VCENTER_RESOURCE_POOL attribute inside a USER_INPUT element, an it would contain a string that represents a list. Let's see an example:

```
USER_INPUTS=[
    VCENTER_RESOURCE_POOL="O|list|Which resource pool you want this VM to run in?␣
↪|TestResourcePool/NestedResourcePool,TestResourcePool|TestResourcePool/
↪NestedResourcePool" ]
```

(continues on next page)

(continued from previous page)

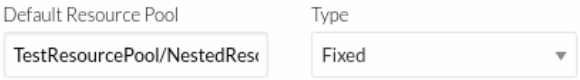
The VCENTER_RESOURCE_POOL has the following elements:

- O: it means that it is optional to select a Resource Pool.
- list: this will be a list shown to users.
- Which resource pool you want this VM to run in?: that's the question that will be shown to users.
- TestResourcePool/NestedResourcePool,TestResourcePool: that's the list of Resource Pool references separeted with commas that are available to the user.
- TestResourcePool/NestedResourcePool: is the default Resource Pool that will be selected on the list.

Note: As we'll see later, the import tools provided by OpenNebula will create the VCENTER_RESOURCE_POOL attribute easily.

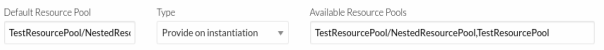
Using Sunstone we have the same actions described for the onevcenter tool.

If we want to specify a Resource Pool we should select Fixed from the Type drop-down menu and introduce the reference under Default Resource Pool:



If we wanted to offer a list to the user:

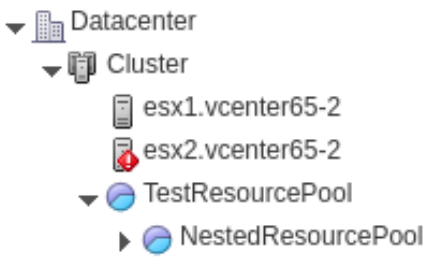
- We would select Provide on Instantiation from the Type drop-down menu.
- We would specify the default value that we want to be selected in the list.
- We would introduce the references of the Resource Pools that we want to include in the list, using a comma to separate values.



Referencing a Resource Pool

The VCENTER_RESOURCE_POOL attribute expects a string containing the name of the Resource Pool. If the Resource Pool is nested, the name of the Resource Pool should be preceeded by slashes and the names of the parent Resource Pools.

For instance, a Resource Pool "NestedResourcePool" nested under "TestResourcePool"



would be represented as "TestResourcePool/NestedResourcePool":

```
VCENTER_RESOURCE_POOL="TestResourcePool/NestedResourcePool"
```

Resource deletion in OpenNebula

There are different behavior of the vCenter resources when deleted in OpenNebula.

The following resources are NOT deleted in vCenter when deleted in OpenNebula:

- VM Templates.
- Networks. Unless OpenNebula has created the port groups and/or switches instead of just consume them.
- Datastores.

The following resource are deleted in vCenter when deleted in OpenNebula:

- Virtual Machines.
- Images. A VMDK or ISO file will be deleted in vCenter unless the VCENTER_IMPORTED attribute is set to YES.

2.13.2 Considerations & Limitations

- **Unsupported Operations**: The following operations are **NOT** supported on vCenter VMs managed by OpenNebula, although they can be performed through vCenter:

Operation	Note
disk snapshots	Only system snapshots are available for vCenter VMs

- **No Security Groups**: Firewall rules as defined in Security Groups cannot be enforced in vCenter VMs.
- Image names cannot contain spaces.
- Detach disks operations are not supported in VMs with system snapshots. Since vCenter snapshots considers disks and are tied to them, disks cannot be removed afterwards.
- vCenter credential password cannot have more than 22 characters.
- If you are running Sunstone using nginx/apache you will have to forward the following headers to be able to interact with vCenter, HTTP_X_VCENTER_USER, HTTP_X_VCENTER_PASSWORD and HTTP_X_VCENTER_HOST (or, alternatively, X_VCENTER_USER, X_VCENTER_PASSWORD and X_VCENTER_HOST). For example in nginx you have to add the following attrs to the server section of your nginx file: (underscores_in_headers on; proxy_pass_request_headers on;).

Snapshot limitations

OpenNebula treats **snapshots** a tad different than VMware. OpenNebula assumes that they are independent, whereas VMware builds them incrementally. This means that OpenNebula will still present snapshots that are no longer valid if one of their parent snapshots are deleted, and thus revert operations applied upon them will fail. The snapshot preserves the state and data of a virtual machine at a specific point in time including disks, memory, and other devices, such as virtual network interface cards so this operation may take some time to finish.

vCenter impose some limitations and its behavior may differ from vCenter 5.5 to 6.5. If you create a snapshot in OpenNebula note the following limitations:

- **It's not a good idea to add or detach disks or nics if you have created a snapshot**. DISKs and NICs elements will be removed from your OpenNebula VM and if you revert to your snapshot, those elements that were added or removed won't be added again to OpenNebula VM and vCenter configuration may not be in sync with OpenNebula's representation of the VM. It would be best to remove any snapshot, perform the detach actions and then create a snapshot again affecting operations.
- If despite the previous point you try to detach a disk while the VM is powered on, OpenNebula will not allow this action. If you detach the disk while the VM is in POWEROFF OpenNebula will remove the DISK element but the disk won't be removed from vCenter.
- You cannot perform the disk save as operation unless the VM is powered off.
- You cannot resize disks.

2.13.3 Configuring

The vCenter virtualization driver configuration file is located in `/etc/one/vcenter_driver.default`. This XML file is home for default values for OpenNebula VM templates and images.

Default values for Virtual Machine attributes will be located inside a TEMPLATE element under a VM element while default values for Images (e.g a representation of a VMDK file) will be located inside a TEMPLATE element under an IMAGE element.

So far the following default values for VM NIC can be set:

Attribute	Description	Values
MODEL	It will specify the network interface card model. By default it is set to vmxnet3.	e1000 e1000e pcnet32 sriovethernetcard vmxnetm vmnet2 vmnet3
INBOUND_AVG_BW	Average bitrate for the interface in kilobytes/second for inbound traffic.	
INBOUND_PEAK_BW	Maximum bitrate for the interface in kilobytes/second for inbound traffic.	
OUTBOUND_AVG_BW	Average bitrate for the interface in kilobytes/second for outbound traffic.	
OUTBOUND_PEAK_BW	Maximum bitrate for the interface in kilobytes/second for outbound traffic.	

Note: You can use the template attribute NIC_DEFAULT/MODEL to set a default nic model into your deployed machines. Note that NIC/MODEL have preference and priority. If you don't explicitly set any of these values vCenter driver will: for OpenNebula managed NICs (managed NICs) get the model from vCenter driver defaults file (/etc/one/vcenter_driver.default), and for vCenter managed NICs (unmanaged NICs) delegate the decision to vCenter (ie. will use the vCenter VM Template definition of the NIC).

Default attributes for VM GRAPHICS:

Attribute	Description	Values
KEYMAP	It will specify the keymap for a remote access through VNC	any keymap code

So far the following default values for an IMAGE can be set:

Attribute	Description	Values
VCENTER_ADAPTER_TYPE	Controller that will handle the image in vCenter	lsiLogic ide busLogic
VCENTER_DISK_TYPE	The vCenter Disk Type of the image.	thin thick eagerZeroedThick
DEV_PREFIX	Prefix for the emulated device the image will be mounted at. By default **sd** is used.	hd sd

It is generally a good idea to place defaults for vCenter-specific attributes. The following is an example:

```
<VCENTER>
    <VM>
        <TEMPLATE>
            <NIC>
                <MODEL>vmxnet3</MODEL>
                <INBOUND_AVG_BW>100</INBOUND_AVG_BW>
            </NIC>
            <GRAPHICS>
                <KEYMAP>US</KEYMAP>
            </GRAPHICS>
        </TEMPLATE>
    </VM>
    <IMAGE>
        <TEMPLATE>
            <DEV_PREFIX>sd</DEV_PREFIX>
            <VCENTER_DISK_TYPE>thin</DISK_TYPE>
            <VCENTER_ADAPTER_TYPE>lsiLogic</ADAPTER_TYPE>
        </TEMPLATE>
    </IMAGE>
</VCENTER>
```

Also you have `/etc/one/vcenter_driver.conf` where you can define the following attributes:

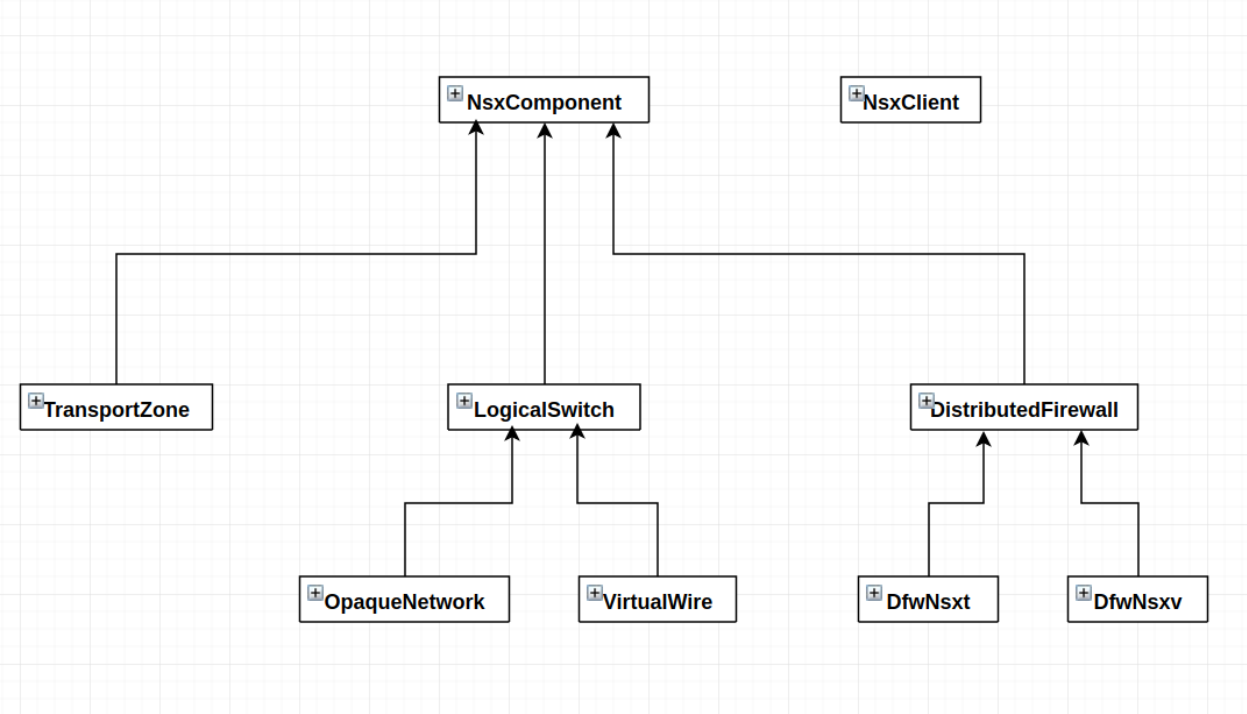
Attribute	Description	Value
vm_poweron_wait_default	Timeout to deploy VMs in vCenter	Integer
debug_information	If you want more error information in vCenter driver	true or false

2.14 NSX Driver

The NSX Driver for OpenNebula enables the interaction with NSX Manager API, to manage its different components, such as logical switches, distributed firewall and so on.

2.14.1 UML diagram

Here is the dependency between classes into the NSX driver.



2.14.2 NSX Integration

Logical switches integration

OpenNebula can manage logical switches using NSX Driver and the hook subsystem. How does it work?

An action to create or delete a logical switch either from Sunstone, CLI or API, generates an specific event.

If there is a hook subscribed to that event, it will execute an action or script.

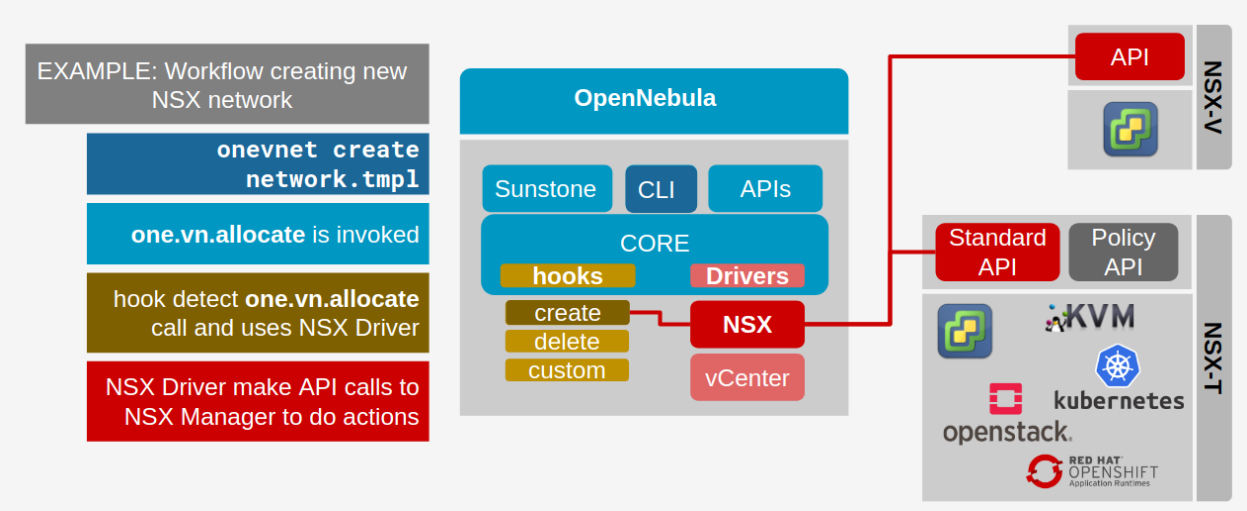
In the case of NSX, the hook will use the NSX Driver to send commands to the NSX Manager API, and will wait to the answer.

When NSX Manager finish the action, will return a response and the hook, based on that response, will end up as success or error.

The process of creating a logical switch is depicted below.

2.14.3 Object references

All NSX object has its reference within the NSX Manager.



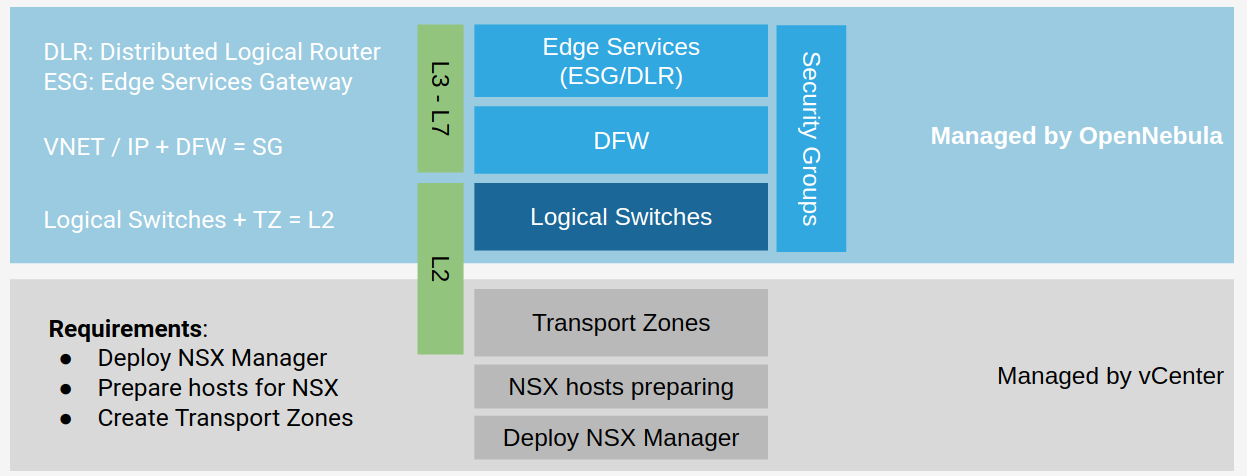
Some NSX objects also has its reference into vCenter Server. This is the case of logical switches, that have its object representation in vCenter.

Here is a table with the components and their reference formats in NSX and vCenter, and also the attributes used in OpenNebula to store that object:

Object	NSX type	ONE Attribute	NSX ref. format	vCenter ref format
Logical Switch (Opaque Network)	NSX-T	NSX_ID	xxxxxxxx-yyyy-zzzz-aaaa-bbbbbbbbbbbb	network-oXXX
Logical Switch (VirtualWire)	NSX-V	NSX_ID	virtualwire-XXX	dvportgroup-XXX
Transport Zone	NSX-T	TZ_ID	xxxxxxxx-yyyy-zzzz-aaaa-bbbbbbbbbbbb	N/A
Transport Zone	NSX-V	TZ_ID	vdnscope-XX	N/A

2.14.4 Managed components

Currently only Logical Switches are supported. In the image below is shown which components are intended to be supported by OpenNebula through its NSX Driver in the near future.


DLR: Distributed Logical Router
ESG: Edge Services Gateway
VNET / IP + DFW = SG
Logical Switches + TZ = L2
L3 - L7
L2
Edge Services (ESG/DLR)
DFW
Logical Switches
Security Groups
Managed by OpenNebula
Requirements:
Deploy NSX Manager
Prepare hosts for NSX
Create Transport Zones
Transport Zones
NSX hosts preparing
Deploy NSX Manager
Managed by vCenter

CHAPTER

THREE

REFERENCES

3.1 Overview

This Chapter contains advanced references to compile OpenNebula from the source code and how to extend the Sunstone functionality.

3.1.1 How Should I Read This Chapter

You should be reading this Chapter if you are trying to build OpenNebula from source code or trying to extend the Sunstone functionality.

This Chapter covers what are the OpenNebula *software dependencies*, how to *compile* OpenNebula and how to *extend Sunstone functionality*.

After this Chapter, if you are interested in extending the OpenNebula functionality, you need to read the *Infrastructure Integration Chapter* to understand how OpenNebula relates with the different datacenter components.

3.1.2 Hypervisor Compatibility

All the Sections of this Chapter applies to all hypervisors.

3.2 Building from Source Code

This page will show you how to compile and install OpenNebula from the sources.

If you want to install it from your package manager, visit the software menu to find out if OpenNebula is included in your official distribution package repositories.

> **Warning:** Do not forget to check the *Building Dependencies* for a list of specific software requirements to build OpenNebula.

3.2.1 Compiling the Software

Follow these simple steps to install the OpenNebula software:

- Download and untar the OpenNebula tarball.
- Change to the created folder and run scons to compile OpenNebula

```
    $ scons [OPTION=VALUE]

the argument expression [OPTION=VALUE] is used to set non-default values for :
```

OPTION	VALUE
sqlite_db	path-to-sqlite-install
sqlite	**no** if you don't want to build sqlite support
mysql	**yes** if you want to build mysql support
xmlrpc	path-to-xmlrpc-install
parsers	**yes** if you want to rebuild flex/bison files
new_xmlrpc	**yes** if you have an xmlrpc-c version >= 1.31
sunstone	**yes** if you want to build sunstone minified files
systemd	**yes** if you want to build systemd support
docker_machine	**yes** if you want to build the docker-machine driver
svncterm	**no** to skip building vnc support for LXD drivers

If the following error appears, then you need to remove the option 'new_xmlrpc=yes' or install xmlrpc-c version >= 1.31:

```
error: 'class xmlrpc_c::serverAbyss::constrOpt' has no member named 'maxConn'
```

- OpenNebula can be installed in two modes: `system-wide`, or in `self-contained` directory. In either case, you do not need to run OpenNebula as root. These options can be specified when running the install script:

```
./install.sh <install_options>
```

Note: To install OpenNebula with the `system-wide` mode you should have super user privileges.

```
$ sudo ./install.sh <install_options>
```

where <*install_options*> can be one or more of:

OPTION	VALUE
-u	user that will run OpenNebula, defaults to user executing install.sh
-g	group of the user that will run OpenNebula, defaults to user executing install.sh
-k	keep configuration files of existing OpenNebula installation, useful when upgrading. This flag should not be set when installing OpenNebula for the first time
-d	target installation directory. If defined, it will specified the path for the **self-contained** install. If not defined, the installation will be performed **system wide**
-c	only install client utilities: OpenNebula cli and ec2 client files
-s	install OpenNebula Sunstone
-p	do not install OpenNebula Sunstone non-minified files
-G	install OpenNebula Gate
-f	install OpenNebula Flow
-r	remove Opennebula, only useful if -d was not specified, otherwise `rm -rf $ONE_LOCATION` would do the job
-l	creates symlinks instead of copying files, useful for development
-h	prints installer help
-e	install Docker Machine plugin

Note:

If you choose the `system-wide` installation, OpenNebula will be installed in the following folders:

- /etc/one
- /usr/lib/one
- /usr/share/docs/one
- /usr/share/one
- /var/lib/one
- /var/lock/one
- /var/log/one
- /var/run/one

By using `./install.sh -r`, dinamically generated files will not be removed.

The packages do a `system-wide` installation. To create a similar environment, create a `oneadmin` user and group, and execute:

```
oneadmin@frontend:~/ $> wget <opennebula tar gz>
oneadmin@frontend:~/ $> tar xzf <opennebula tar gz>
oneadmin@frontend:~/ $> cd opennebula-x.y.z
oneadmin@frontend:~/opennebula-x.y.z/ $> scons -j2 mysql=yes syslog=yes
[ lots of compiling information ]
scons: done building targets.
oneadmin@frontend:~/opennebula-x.y.z $> sudo ./install.sh -u oneadmin -g oneadmin
```

3.2.2 Ruby Dependencies

Ruby version needs to be:

- **ruby** >= 1.8.7

Some OpenNebula components need ruby libraries. Some of these libraries are interfaces to binary libraries and the development packages should be installed in your machine. This is the list of the ruby libraries that need a development package:

- **sqlite3**: sqlite3 development library
- **mysql**: mysql client development library
- **curb**: curl development library
- **nokogiri**: expat development librarie
- **xmlparse**: libxml2 and libxslt development libraries

You will also need ruby development package to be able to compile these gems.

We provide a script to ease the installation of these gems. it is located in `/usr/share/one/install_gems` (system-wide mode). It can be called with the components you want the gem dependencies to be installed. Here are the options:

- **optional**: libraries that make CLI and OCA faster
- **quota**: quota system

- **sunstone**: sunstone graphical interface
- **cloud**: ec2 and occi interfaces
- **ozones_client**: CLI of ozones
- **ozones_server**: server part of ozones, both mysql and sqlite support
- **ozones_server_sqlite**: ozones server, only sqlite support
- **ozones_server_mysql**: ozones server, only mysql support
- **acct**: accounting collector, both mysql and sqlite support
- **acct_sqlite**: accounting collector, only sqlite support
- **acct_mysql**: accounting collector, only mysql support

The tool can be also called without parameters and all the packages will be installed.

For example, to install only requirements for sunstone and ec2 interfaces you'll issue:

```
oneadmin@frontend: $> ./install_gems sunstone cloud
```

3.2.3 Building Python Bindings from source

In order to build the OpenNebula python components it is required to install pip package manager and following pip packages:

Build Dependencies:

- **generateds**: to generate the python OCA
- **pdoc**: to generate the API documentation
- **setuptools**: to generate python package
- **wheel**: to generate the python package

Run Dependencies:

- **aenum**: python OCA support
- **dicttoxml**: python OCA support
- **feature**: python OCA support
- **lxml**: python OCA support
- **six**: python OCA support
- **tblib**': python OCA support
- **xmltodict**: python OCA support

To build run following:

```
root@frontend:~/ $> cd src/oca/python
root@frontend:~/ $> make
root@frontend:~/ $> make dist
root@frontend:~/ $> make install
```

3.2.4 Building Sunstone from Source

Please check the *Sunstone Development guide* for detailed information

3.2.5 Building Docker Machine Plugin from Source

Requirements

- **Go >= 1.9**
- **dep** (https://github.com/golang/dep)

Scons includes an option to build the Docker Machine Plugin using the *docker_machine* option:

```
scons docker_machine=yes
```

Once you have builded you can install it running the install.sh with the *-e* option.

3.2.6 Configure sudo for oneadmin

`oneadmin` user, both on frontend and nodes, needs to run several commands under a privileged user via `sudo`. When installing the OpenNebula from official packages, the necessary configuration is part of the `opennebula-common` package. When installing from the source, you have to ensure the proper `sudo` configuration enables following commands to the `oneadmin`.

Section	Commands
networking	ebtables, iptables, ip6tables, ip, ipset
LVM	lvcreate, lvremove, lvs, vgdisplay, lvchange, lvscan
Open vSwitch	ovs-ofctl, ovs-vsctl
Ceph	rbd
LXD	lsblk, losetup, mount, umount, kpartx, qemu-nbd, mkdir, blkid, e2fsck, resize2fs, xfs_growfs, rbd, rbd-nbd /snap/bin/lxd, /usr/bin/catfstab
HA	systemctl start opennebula-flow, systemctl stop opennebula-flow, systemctl start opennebula-gate, systemctl stop opennebula-gate, service opennebula-flow start, service opennebula-flow stop, service opennebula-gate start, service opennebula-gate stop, arping

Each command has to be specified with the absolute path, which can be different for each platform. Commands are started on background, `sudo` needs to be configured **not to require real tty** and any password for them.

The main sudoers file suitable for the front-end with distribution-specific command paths can be created by the sudoers generator sudo_commands.rb. You only need to ensure that all listed commands are already installed on your system so that the generator can detect their filesystem location. Generated sudo commands aliases must be enabled additionally.

Example configuration

You can put following `sudo` configuration template into `/etc/sudoers.d/opennebula` and replace example commands `/bin/true` and `/bin/false` with comma separated list of commands listed above, with the absolute path specific for your platform.

```
Defaults:oneadmin !requiretty
Defaults:oneadmin secure_path = /sbin:/bin:/usr/sbin:/usr/bin

oneadmin ALL=(ALL) NOPASSWD: /bin/true, /bin/false
```

Qemu configuration

Qemu should be configured to not change file ownership. Modify `/etc/libvirt/qemu.conf` to include `dynamic_ownership = 0`. To be able to use the images copied by OpenNebula, change also the user and group below the dynamic_ownership setting"

LXD configuration

Add oneadmin to the lxd and libvirt group:

```
usermod -aG lxd oneadmin
usermod -aG libvirt oneadmin
```

If you plan to user qcow2 images on LXD, then you should load the **nbd** kernel module.

```
modprobe nbd
```

3.2.7 Starting OpenNebula

Setup authentication file.

```
echo '$USER:password' > ~/.one/one_auth
```

Start the opennebula server

```
one start
```

Check it worked

```
oneuser show
USER 0 INFORMATION
ID              : 0
NAME            : oneadmin
GROUP           : oneadmin
PASSWORD        : 4478db59d30855454ece114e8ccfa5563d21c9bd
AUTH_DRIVER     : core
ENABLED         : Yes

TOKENS

USER TEMPLATE
TOKEN_PASSWORD="f99aab65e58162dc83a0fae59bec074a935c9a68"

VMS USAGE & QUOTAS

VMS USAGE & QUOTAS - RUNNING

DATASTORE USAGE & QUOTAS
```

(continues on next page)

(continued from previous page)

```
NETWORK USAGE & QUOTAS

IMAGE USAGE & QUOTAS
```

3.3 Build Dependencies

This page lists the **build** dependencies for OpenNebula.

If you want to install it from your package manager, visit the software menu to find out if OpenNebula is included in your official distribution package repositories.

- **g++** compiler (>= 4.0)
- **xmlrpc-c** development libraries (>= 1.06)
- **scons** build tool (>= 0.98)
- **sqlite3** development libraries (if compiling with sqlite support) (>= 3.6)
- **mysql** client development libraries (if compiling with mysql support) (>= 5.1)
- **libxml2** development libraries (>= 2.7)
- **libvncserver** development libraries (>= 0.9)
- **openssl** development libraries (>= 0.9.8)
- **ruby** interpreter (>= 1.8.7)

3.3.1 Ubuntu 18.04

- **bash-completion**
- **bison**
- **debhelper (>= 7.0.50~)**
- **default-jdk**
- **flex**
- **javahelper (>= 0.32)**
- **kpartx**
- **libmysql++-dev**
- **libsqlite3-dev**
- **libssl-dev**
- **libsystemd-dev**
- **libws-commons-util-java**
- **libxml2-dev**
- **libxmlrpc3-client-java**
- **libxmlrpc3-common-java**

- **libxmlrpc-c++8-dev**
- **libxslt1-dev**
- **libcurl4-openssl-dev**
- **libcurl4**
- **libvncserver-dev**
- **ruby**
- **scons**

3.3.2 Ubuntu 16.04

- **bash-completion**
- **bison**
- **debhelper (>= 7.0.50~)**
- **default-jdk**
- **flex**
- **javahelper (>= 0.32)**
- **kpartx**
- **libmysql++-dev**
- **libsqlite3-dev**
- **libssl-dev**
- **libsystemd-dev**
- **libws-commons-util-java**
- **libxml2-dev**
- **libxmlrpc3-client-java**
- **libxmlrpc3-common-java**
- **libxslt1-dev**
- **libcurl4-openssl-dev**
- **libvncserver-dev**
- **ruby**
- **scons**

3.3.3 Ubuntu 14.04

- **bash-completion**
- **bison**
- **debhelper (>= 7.0.50~)**
- **default-jdk**
- **flex**

- **javahelper (>= 0.32)**
- **libmysql++-dev**
- **libsqlite3-dev**
- **libssl-dev**
- **libws-commons-util-java**
- **libxml2-dev**
- **libxmlrpc3-client-java**
- **libxmlrpc3-common-java**
- **libxslt1-dev**
- **libcurl4-openssl-dev**
- **libvncserver-dev**
- **ruby**
- **scons**
- **libxmlrpc-c++8-dev**

3.3.4 Debian 9

- **bison**
- **default-jdk**
- **flex**
- **javahelper**
- **libmysql++-dev**
- **libsqlite3-dev**
- **libssl-dev**
- **libsystemd-dev**
- **libws-commons-util-java**
- **libxml2-dev**
- **libxmlrpc-c++8-dev**
- **libxmlrpc3-client-java**
- **libxmlrpc3-common-java**
- **libxslt1-dev**
- **libvncserver-dev**
- **ruby**
- **scons**

3.3.5 CentOS 7

- **gcc-c++**
- **java-1.7.0-openjdk-devel**
- **libcurl-devel**
- **libxml2-devel**
- **mysql-devel**
- **openssh**
- **openssl-devel**
- **pkgconfig**
- **ruby**
- **scons**
- **sqlite-devel**
- **systemd-devel**
- **xmlrpc-c**
- **xmlrpc-c-devel**
- **libvncserver-devel**

3.3.6 CentOS 6

- **gcc-c++**
- **libcurl-devel**
- **libxml2-devel**
- **xmlrpc-c-devel**
- **openssl-devel**
- **mysql-devel**
- **openssh**
- **pkgconfig**
- **ruby**
- **scons**
- **sqlite-devel**
- **xmlrpc-c**
- **java-1.7.0-openjdk-devel**
- **libvncserver-devel**

3.3.7 Arch

They are listed in this PKGBUILD.

3.4 Sunstone Development

In OpenNebula 5.0 the graphical interface code, Sunstone, was redesigned and modularized to improve the code readability and ease the task of adding new components. Now, each component (tab, panel, form...) is defined in a separate module using requirejs and HTML code is now defined in separate files using Handlebars templates. External libraries are handled using Bower, a Javascript package manager. Zurb Foundation is used for the styles and layout of the web and additional CSS styles are added using SASS. Grunt is used as a tasker to automate the different processes to generate the optimized files.

3.4.1 RequireJS

requirejs allows you to define blocks of functionality in different modules/files and then load and reuse them in other modules.

This is an example of the images-tab files layout:

```
images-tab.js
images-tab/
  actions.js
  buttons.js
  datatable.js
  dialogs/
    ...
  panels/
    ...
```

And how the different modules are used in the images-tab file:

```
/* images-tab.js content */

define(function(require) {
  var Buttons = require('./images-tab/buttons');
  var Actions = require('./images-tab/actions');
  var Table = require('./images-tab/datatable');

  var _dialogs = [
    require('./images-tab/dialogs/clone')
  ];

  var _panels = [
    require('./images-tab/panels/info'),
    require('./images-tab/panels/vms'),
    require('./images-tab/panels/snapshots')
  ];
})
```

The optimization tool provided by requirejs is used to group multiple files into a single minified file (dist/main.js). The options for the optimization are defined in the Gruntfile.js file using the r.js plugin for Grunt.

3.4.2 Handlebars

Handlebars provides the power necessary to let you build semantic templates and is largely compatible with Mustache templates.

```
<div id="comments">
  {{#each comments}}
  <h2><a href="/posts/{{../permalink}}#{{id}}">{{title}}</a></h2>
  <div>{{body}}</div>
  {{/each}}
</div>
```

The integration between Handlebars and requirejs is done using the Handlebars plugin for requirejs, that allows you to use the templates just adding a prefix to the require call

```
var TemplateHTML = require('hbs!./auth-driver/html');
return TemplateHTML({
  'dialogId': this.dialogId,
  'userCreationHTML': this.userCreation.html()
});
```

Additional helpers can be defined just creating a new file inside the `app/templates/helpers` folder. These helpers will be available for any template of the app.

```
{{#isTabActionEnabled "vms-tab" "VM.attachdisk"}}
<span class="right">
  <button id="attach_disk" class="button tiny success right radius">
    {{tr "Attach disk"}}
  </button>
</span>
{{/isTabActionEnabled}}
```

3.4.3 SASS & Foundation

The Zurb Foundation framework is used for the layout of the app. It provides a powerful grid system and different nifty widgets such as tabs, sliders, dialogs...

The Zurb Foundation configuration parameters are defined in the `app/scss/settings.scss` file and new styles for the app can be added in the `app/scss/app.scss` file. After modifying these files, the app.css and app.min.css files must be generated as explained in the following section.

3.4.4 Modifying JS & CSS files

Sunstone can be run in two different environments:

- Production, using the minified css `css/app.min.css` and javascript `dist/main.js` files.
- Development, using the non minified css `css/app.css` and javascript files `app/main.js`. Note that each file/module will be retrieved in a different HTTP request and the app will take longer to start, therefore it is not recommended for production environments

By default Sunstone is configured to use the minified files, therefore any change in the source code will not apply until the minified files are generated again. But you can set the `env` parameter in sunstone-server.conf to `dev` to use the non minified files and test your changes.

After testing the changes, the minified files can be generated by running the `grunt requirejs` task or the `scons sunstone=yes` command as explained in the following section. It is recommended to change again the `env` parameter in sunstone-server.conf to `prod` and test again the changes.

```
sunstone/
  public/
    app/                    # JS sources
    bower_components/       # External libraries
    css/                    # CSS optimized files
    dist/                   # JS optimized files
    node_modules/           # Development dependencies
    scss/                   # CSS sources
    bower.json              # List of external libraries
    Gruntfile.js            # Tasks to optimize files
    package.json            # List of dev dependencies
  routes/                   # Custom routes for Sunstone Server
```

Sunstone Development Dependencies

1. Install nodejs and npm
2. Install the following npm packages:

```
sudo npm install -g bower
sudo npm install -g grunt
sudo npm install -g grunt-cli
```

3. Move to the Sunstone public folder and run:

```
npm install     # Dependencies defined in package.json
bower install   # Dependenciees define in bower.json
```

Warning: In order to run npm and bower commands `git` is required

Building minified JS and CSS files

Scons includes an option to build the minified JS and CSS files. Sunstone development dependencies must be installed before running this command.

```
scons sunstone=yes
```

Or you can do this process manually by running the following commands:

Run the following command to generate the app.css file in the css folder, including any modification done to the `app/scss/app.scss` and `app/scss/settings.scss` files:

```
grunt sass
```

Run the following command to generate the minified files in the dist folder, including any modification done to the js files and the app.min.css in the css folder, based on the app.css file generated in the previous step:

```
grunt requirejs
```

These are the files generated by the `grunt requirejs` command:

```
css
    app.min.css
dist
    login.js login.js.map main-dist.js main.js.map
console
    spice.js spice.js.map vnc.js vnc.js.map
```

Warning: If the following error appears when running scons sunstone=yes or any of the grunt commands, you may have skip one step, so move to the Sunstone public folder and run 'bower install'

```
Running "sass:dist" (sass) task
>> Error: File to import not found or unreadable: util/util
...
>>         on line 43 of scss/_settings.scss
>> >> @import 'util/util';
>>    ^
Warning:  Use --force to continue.
```

Install.sh

By default the install.sh script will install all the files, including the non-minified ones. Providing the -p option, only the minified files will be installed.

The script generates a symbolic link **main.js** pointing to `VAR_LOCATION/sunstone/main.js`. This file has been generated the first time that Sunstone starts, joining the base of Sunstone and the active addons.

3.4.5 Adding Custom Tabs

New tabs can be included following these steps:

- Add your code inside the `app` folder. The tab must be provided as a module.
- Include the new tab as a dependency in the `app/main.js` file for the `app` module.

```
shim: {
  'app': {
    deps: [
      'tabs/provision-tab',
      'tabs/dashboard-tab',
      'tabs/system-tab',
      ...
      'tabs/mycustom-tab'
    ]
  },
```

- Include the tab configuration inside the different Sunstone views `/etc/one/sunstone-views/(admin|user|...).yaml`

```
enabled_tabs:
  - dashboard-tab
  - system-tab
  ...
  - mycustom-tab
```

(continues on next page)

(continued from previous page)

```
tabs:
  mycustom-apps-tab:
      panel_tabs:
          myscustom_info_tab: true
      table_columns:
          - 0         # Checkbox
          - 1         # ID
          - 2         # Name
      actions:
          MyCustom.create: true
          MyCustom.refresh: true
```

- Generate the minified files including the new tab by running the `grunt requirejs` command.

You can see an example of external tabs and custom routes for AppMarket in its own Github repository

3.4.6 Custom Routes for Sunstone Server

OpenNebula Sunstone server plugins consist of a set files defining custom routes. Custom routes will have priority over default routes and allow administrators to integrate their own custom controllers in the Sunstone Server.

Configuring Sunstone Server Plugins

It is very easy to enable custom plugins:

1. Place your custom routes in the `/usr/lib/one/sunstone/routes` folder.
2. Modify `/etc/one/sunstone-server.conf` to indicate which files should be loaded, as shown in the following example:

```
# This will load ''custom.rb'' and ''other.rb'' plugin files.
:routes:
    - custom
    - other
```

Creating Sunstone Server Plugins

Sunstone server is a Sinatra application. A server plugin is simply a file containing one or several custom routes, as defined in sinatra applications.

The following example defines 4 custom routes:

```
get '/myplugin/myresource/:id' do
    resource_id = params[:id]
    # code...
end

post '/myplugin/myresource' do
    # code
end

put '/myplugin/myresource/:id' do
    # code
end
```

(continues on next page)

(continued from previous page)

```
del '/myplugin/myresource/:id' do
    # code
end
```

Custom routes take preference over Sunstone server routes. In order to ease debugging and ensure that plugins are not interfering with each other, we recommend however to place the routes in a custom namespace (`myplugin` in the example).

From the plugin code routes, there is access to all the variables, helpers, etc. which are defined in the main sunstone application code. For example:

```
opennebula_client = $cloud_auth.client(session[:user])
sunstone_config = $conf
logger.info("New route")
vm3_log = @SunstoneServer.get_vm_log(3)
```

3.4.7 ESLint

Install ESLint:

```
sudo npm install -g eslint
```

After the installation you can initialize ESLint with your own rules or use OpenNebula's configuration:

1. Use the command `eslint --init` to create your own *.eslintrc.json* with your personal configuration.

   or

2. Manually create the *.eslintrc.json* and copy/paste the following code:

`one/src/sunstone/public/.eslintrc.json`

```
{
  "env": {
      "browser": true,
      "es6": true
  },
  "parserOptions": {
      "sourceType": "module"
  },
  "rules": {
      "linebreak-style": [
          "error",
          "unix"
      ],
      "quotes": [
          "error",
          "double"
      ],
      "semi": [
          "error",
          "always"
      ],
      "eqeqeq": 2,
      "no-trailing-spaces": [
```

(continues on next page)

(continued from previous page)

```
            "error"
        ]
        //new rules here
    }
}
```

Note: The usage of ESLint is not mandatory but we recomend our contributors to use it, to be sure that the code is standardiced.

More information about ESlint project.