OpenNebula.org

OpenNebula 4.14 Release Notes

Release 4.14.0

OpenNebula Project

September 25, 2015

CHAPTER

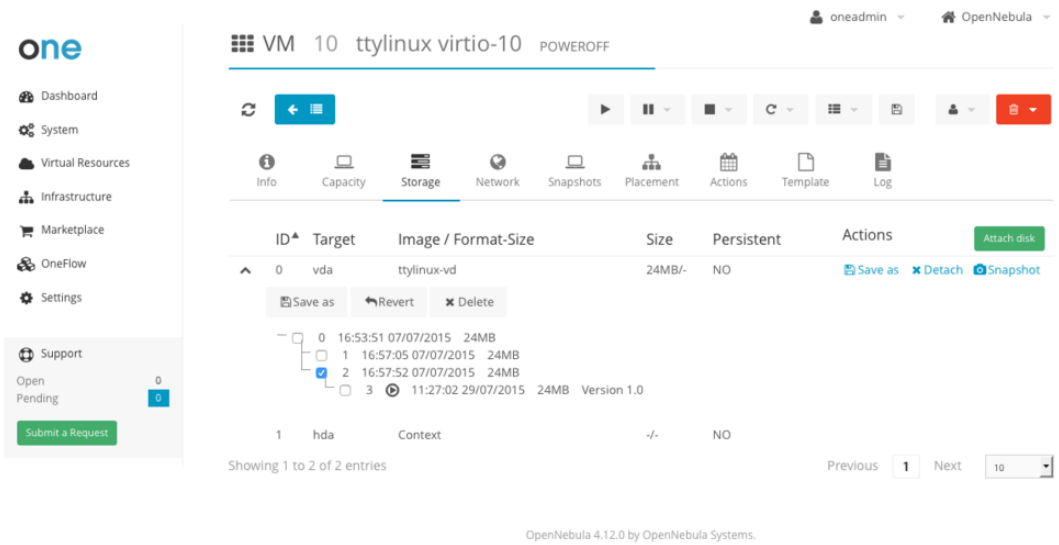
ONE

RELEASE NOTES 4.14

1.1 What's New in 4.14

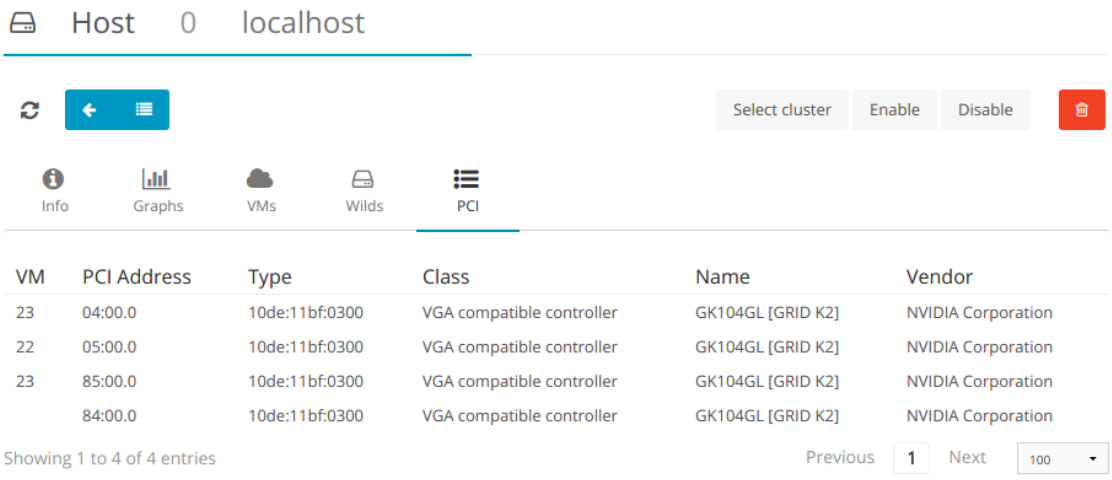
OpenNebula 4.14 (Great A'Tuin) ships with several improvements in different subsystems and components. The Sunstone interface has been completely refactored, for maintenance and performance reasons. Expect major improvements in Sunstone from now on. Also, we are sure you will like the subtle changes in the look and feel.

Several major features have been introduced in Great A'Tuin. One of the most interesting for cloud users and administrators is the ability to create and maintain a tree of VM disks snapshots, in this version for Ceph and qcow2 backends. Now VM disks can be reverted to a previous state at any given time, and they are preserved in the image if it is persistent in the image datastore. For instance, you can attach a disk to a VM, create a snapshot, detach it and attach it to a new VM, and revert to a previous state. Very handy, for instance, to keep a working history of datablocks that can contain dockerized applications.



Another major feature is the ability to resize an existing disk, for all the supported OpenNebula backends. If your VM needs more disk space than the one provided by the images used for its disk, you can now set a new size prior to instantiate the VM, OpenNebula will grow the disk and the guest OS will adapt the filesystem to the now bigger disk at boot time. The disk space is not an issue anymore. Moreover, cloud admins can now keep track of disk usage in their infrastructure since now disk consumption is taken into account for quotas, accounting and showback; and this calculations includes these two features (snapshot + resizing).

To support HPC oriented infrastructures based on OpenNebula, 4.14 also enables the consumption of raw GPU devices existing on a physical host from a Virtual Machine. There is no overcommitment possible nor sharing of GPU devices among different Virtual Machines, so a new type of consumable has been defined in OpenNebula and taken into account by the scheduler. VMs can now request a GPU, and if OpenNebula finds one free resource of type GPU available, it will set up the VM with PCI passthrough access to the GPU resource, enabling applications to get the performance boost of the direct access to a GPU card.



The ability to save VMs into VM Templates for later use is another feature that must be highlighted in this release. This new operation is accessible both from the cloud view and the admin Sunstone view, as well as from the command line interface. Another great improvement for cloud admins is a much better state management of VMs. It is now possible to recover VMs from failed state instructing OpenNebula to take the last action as success, to retry it or to make it fail gracefully, to recover for instance from failed migrations.

OpenNebula users managing vCenter infrastructures will also benefit from this new version. VM importing workflow

has been greatly improved through Sunstone, making it easier to import your existing workload into OpenNebula. Moreover, 4.14 adds the possibility to instruct OpenNebula whether or not it should save the disks and a very important contextualisation improvement now allows to directly pass scripts to be executed in boot time to vCenter VMs, increasing the flexibility in VM customisation from OpenNebula in vCenter.

There are many other improvements in 4.14, like a more flexible definition of context network attributes, the ability to import running VMs not launched by OpenNebula from all the supported hypervisors (including the hybrid ones, for instance now it is possible to manage through OpenNebula Azure, SoftLayer and EC2 VMs launched through their respective management portals); the possibility to cold attach disks and network interfaces to powered off machines (which complements the hot attach functionality), improvements in accounting and showback to keep track of disk usage, better logging in several areas, the ability to pass scripts to VMs for guest OS customization, and many others. A great effort was put in this release to help build and maintain robust private, hybrid and public clouds with OpenNebula.

This OpenNebula release is named after Great A'Tuin, the Giant Star Turtle (of the fictional species Chelys galactica) who travels through the Discworld universe's space, carrying four giant elephants who in turn carry the Discworld. Allegedly, it is "the only turtle ever to feature on the Hertzsprung–Russell diagram."

Want to take OpenNebula 4.14 for a test drive? Use one of the SandBoxes to try out OpenNebula in no time, or proceed to the Quick Start guides. Great A'Tuin is considered to be a stable release and as such, and update is available in production environments.

In the following list you can check the highlights of OpenNebula 4.14. (a detailed list of changes can be found here):

1.1.1 OpenNebula Core

The OpenNebula Core handles the abstractions that allows to orchestrate the DC resources. In this release, the following additions and improvements are present:

- **Better logging of error messages**, more information now present in the logs to better debug errors.
- **Support for GPU consumables**, giving the ability to give exlcusive PCI passthrough access to VMs to GPU cards, for HPC computing.
- **Improved VM recovery and lifecycle flexibility**, thanks to new state transitions, like for instance recover failed VMs back to running state, cancel deferred snapshots.
- **New maintenance operations**, using cold migration now also lets switch between system datastores. This can be achived both from the CLI and Sunstone.
- **Running VMs can now be imported in all hypervisors**, not only in vCenter. This operation is available through a new WILDS tab in the hosts.
- **Better support for poweroff state**, with for instance the ability of cold disk and NIC attaching.
- **Saving VMs for latter use**, introducing the ability to clone a VM in the poweroff state into a VM template that can be instantiated latter on.
- **More administration flexibility**, with the ability to update host drivers.
- **Improved history logging**, accounting records are also created when the Virtual Machine has a disk/nic attached or detached.
- **Flexible default auth driver definition**, now it can be set in the core configuration file.

New perks also for developers:

- **More robust API**, with the addition of locks at the core level in the document pools, now you can use the core to synchronize operations.

1.1.2 OpenNebula Drivers :: Networking

OpenNebula networking is getting better and better:

- **Host housekeeping**, cleaning VXLAN devices when no VMs are running in the hypervisor.
- **Set Maximum Transmission Unit**, from the network templates in the hypervisor through the 802.1q drivers.

1.1.3 OpenNebula Drivers :: Storage

Exciting new features in the storage subsystem:

- **New disk snapshot capabilities**, now it is possible to snapshot a disk from within OpenNebula and keep a tree of snapshots in the VM and back in the image datastore, reverting (or flattening) at any moment to any snapshot in the tree. If the VM disk where the snapshot is taken is a persistent image, the *snapshots will be persisted back into the image datastore* <*img_guide_snapshots*>. Different backends (like ceph and qcow2) are supported.
- **Disk snapshots in VM running state**, for qcow2 backends.
- **Disk resizing**, grow a VM disk at instantiation time on your VM while conforming with your quotas and being noted down for accounting.

1.1.4 OpenNebula Drivers :: Virtualization

- **Get the real and virtual usage for disks**, file based storage not always use the maximum virtual size of the disk. (for example qcow2 or sparse raw files). Improvements in monitoring take now care of this reporting.
- **Running VMs support** , ability to import VMs running in hypervisors (all of them now supported, even the hybrids) that have not being launched by OpenNebula.
- **Spice support for more hypervisors**, now supported as well in XEN.
- **Control how disks are managed in vCenter**, through a new VM template variable. Protect users data against accidental deletions.

1.1.5 Scheduler

- **Better logging**, now is easier to understand what is going on in the scheduler
- **Control System DS deployment with ACL rules**, the scheduler (and core) has been update to enforce access rights on system datastores, checking that the user can access the System DS. This is useful to implement different allocation policies and VDC-based provision schemes.

1.1.6 Sunstone

Sunstone has been completely refactored, in order to make it easier to maintain and to improve its performance. We hope you like the subtle look and feel changes as well. In addition:

- **Improvements in view selector**, now views can be selected easier and names can be customized.
- **Better user preferences support**, the number of elements displayed in the datatables are remembered per user.
- **Improvements in usability**, to avoid errors, Sunstone now disables VM actions depending on the current state.
- **Ability to save VMs as templates**, for later use. Saved VMs can have now more than one disk.

1.1.7 Contextualization

Contextualization improvements are also present:

- **Added ability to run arbitrary script**, to help customize guest OS using the START_SCRIPTS and START_SCRIPTS_BASE64 new attributes.
- **More flexible network attributes contextualization**, with the ability of overriding parameters from the network in the Context section.

1.1.8 Command Line Interface

The CLI has not been neglected in this release, to offer all the functionality developed and also to improve several aspects:

- **Default columns for the output reviewed**, to maximize the usefulness of the cli output. For instance, now the IP is shown in the output of onevm list, and the output of the leases table in the onevnet show command has been improved to fit in the owner information.
- **Context shown as another image**, so the target for instance of the context CDROM can be easily found.
- **Better logging and feedback**, for instance for the onedb fsck tool and in onevm help message. Moreover, the onedb upgrade + fsck now save the version of the DB when it backs it up.
- **Ability to import wild VMs**, using the the new onehost importvm command. Also, now onehost sync is disallowed from root accounts to avoid permissions problems.

1.2 Features

This section describes the **detailed features and functionality of OpenNebula** for the management of private clouds and datacenter virtualization(*). It includes links to the different parts of the documentation and the web site that provide extended information about each feature. We also provide a summarized table of key features.

1.2.1 Powerful User Security Management

- Secure and efficient Users and Groups Subsystem for authentication and authorization of requests with complete functionality for user management: create, delete, show...
- Pluggable authentication and authorization based on passwords, ssh rsa keypairs, X509 certificates, LDAP or Active Directory
- Special authentication mechanisms for SunStone (OpenNebula GUI) and the Cloud Services (EC2)
- Login token functionality to password based logins
- Authorization framework with fine-grained ACLs that allows multiple-role support for different types of users and administrators, delegated control to authorized users, secure isolated multi-tenant environments, and easy resource (VM template, VM image, VM instance, virtual network and host) sharing

1.2.2 Advanced Multi-tenancy with Group Management

- Administrators can groups users into organizations that can represent different projects, division...
- Each group have configurable access to shared resources so enabling a multi-tenant environment with multiple groups sharing the same infrastructure

- Configuration of special users that are restricted to public cloud APIs (EC2)
- Complete functionality for management of groups: create, delete, show...
- Multiple group support, with the ability to define primary and secondary groups.

1.2.3 On-demand Provision of Virtual Data Centers

- A VDC is a fully-isolated virtual infrastructure environment where a Group of users, optionally under the control of the VDC admin, can create and manage compute and storage capacity.
- User Groups can be assigned one or more resource providers. Resource providers are defined as a cluster of servers, virtual networks, datastores and public clouds for cloud bursting in an OpenNebula zone. Read more in the Users and Groups Management Guide.
- A special administration group can be defined to manage specific aspects of the group like user management or appliances definition. Read more in the Managing Users and Groups guide.
- Sunstone views for new groups can be dynamically defined without the need of modifying the Sunstone configuration files. More information in the Sunstone Views guide.
- Groups can now be tagged with custom attributes. Read more in the Managing Users and Groups guide.

1.2.4 Advanced Control and Monitoring of Virtual Infrastructure

- Image Repository Subsystem with catalog and complete functionality for VM image management: list, publish, unpublish, show, enable, disable, register, update, saveas, delete, clone...
- Template Repository Subsystem with catalog and complete functionality for VM template management: add, delete, list, duplicate...
- Full control of VM instance life-cycle and complete functionality for VM instance management: submit, deploy, migrate, livemigrate, reschedule, stop, save, resume, cancel, shutdown, restart, reboot, delete, monitor, list, power-on, power-off,...
- Advanced functionality for VM dynamic management like system and disk snapshotting (including hot disk snapshotting, preserving them back into the image datastore if the image is persistent), capacity resizing, or NIC hotplugging
- Programmable VM operations, so allowing users to schedule actions
- Volume hotplugging to easily hot plug a volatile disk created on-the-fly or an existing image from a Datastore to a running VM
- Advanced network virtualization capabilities with traffic isolation, address reservation, flexible defintion of address ranges to accommodate any address distribution, definition of generic attributes to define multi-tier services consisting of groups of inter-connected VMs, and complete functionality for virtual network management to interconnect VM instances: create, delete, monitor, list...
- IPv6 support with definition site and global unicast addresses
- Configurable system accounting statistics to visualize and report resource usage data, to allow their integration with chargeback and billing platforms, or to guarantee fair share of resources among users
- Showback capabilities to define cost associated to CPU/hours and MEMORY/hours per VM Template.
- Tagging of users, VM images and virtual networks with arbitrary metadata that can be later used by other components
- User defined VM tags to simplify VM management and to store application specific data

- Plain files datastore to store kernels, ramdisks and files to be used in context. The whole set of OpenNebula features applies, e.g. ACLs, ownership...
- PCI passthrough available for VMs that need consumption of raw GPU devices existing on a physical host.
- Disk resizing, grow a VM disk at instantiation time on your VM while conforming with your quotas and being noted down for accounting.
- Ability to import VMs running in hypervisors (even the hybrids) that have not being launched by OpenNebula.
- Save arbitrarily complex VMs as templates for later use.

1.2.5 Complete Virtual Machine Configuration

- Complete definition of VM attributes and requirements
- VM attributes can be provided by the user when the template is instantiated
- Support for automatic configuration of VMs with advanced contextualization mechanisms
- Cloud-init support
- Hook Manager to trigger administration scripts upon VM state change
- Wide range of guest operating system including Microsoft Windows and Linux
- Flexible network defintion
- Security Groups to define firewall rules and apply them to Virtual Machines

1.2.6 Advanced Control and Monitoring of Physical Infrastructure

- Configurable to deploy public, private and hybrid clouds
- Host Management Subsystem with complete functionality for management of physical hosts: create, delete, enable, disable, monitor, list...
- Dynamic creation of clusters as a logical set of physical resources, namely: hosts, networks and data stores, within each zone
- Highly scalable and extensible built-in monitoring subsystem

1.2.7 Broad Commodity and Enterprise Platform Support

- Hypervisor agnostic Virtualization Subsystem with broad hypervisor support (Xen, KVM, VMware ESX and VMware vCenter), centralized management of environments with multiple hypervisors, and support for multiple hypervisors within the same physical box
- vCenter support with automatic import tool of existing VMware resources, including existing running VMs, network management and awareness of the presence of ESX hosts behind vCenter
- Support for GPU consumables*, giving the ability to give exlcusive PCI passthrough access to VMs to GPU cards, for HPC computing.
- Storage Subsystem with support for multiple data stores to balance I/O operations between storage servers, or to define different SLA policies (e.g. backup) and performance features for different VM types or users
- Storage Subsystem supporting any backend configuration with different datastore types: file system datastore, to store disk images in a file form and with image transferring using ssh or shared file systems (NFS, GlusterFS, Lustre...), LVM to store disk images in a block device form, Ceph for distributed block device including RBD

format 2, and VMware datastore specialized for the VMware hypervisor that handle the vmdk format and with support for VMFS

- Advanced disk snapshot capabilities with different backends
- Flexible Network Subsystem with integration with Ebtable, Open vSwitch, 802.1Q tagging and VXLAN

1.2.8 Distributed Resource Optimization

- Powerful and flexible requirement/rank matchmaker scheduler providing automatic initial VM placement for the definition of workload and resource-aware allocation policies such as packing, striping, load-aware, affinity-aware...
- Advanced requirement expressions with cluster attributes for VM placement, affinity policies, any host attribute for scheduling expressions, and scheduler feedback through VM tags
- Powerful and flexible requirement/rank matchmaker scheduler for storage load balancing to distribute efficiently the I/O of the VMs across different disks, LUNs or several storage backends
- Resource quota management to allocate, track and limit computing, storage and networking resource utilization
- Support for cgroups on KVM to enforce VM CPU usage as described in the VM Template

1.2.9 Centralized Management of Multiple Zones

- Federation of multiple OpenNebula zones for scalability, isolation or multiple-site support
- Users can seamlessly provision virtual machines from multiple zones with an integrated interface both in Sunstone and CLI.
- A new tool set has been developed to upgrade, integrate new zones and import existing zones into an OpenNebula federation. Read more in the Federation Configuration guide.
- Integrated zone management in OpenNebula core. Read more about this in the Data Center Federation guide.
- Redesigned data model to minimize replication data across zones and to tolerate large latencies. Read more about this in the Data Center Federation guide.
- Complete functionality for management of zones: create, delete, show, list...

1.2.10 High Availability

- Persistent database backend with support for high availability configurations
- Configurable behavior in the event of host, VM, or OpenNebula instance failure to provide an easy to use and cost-effective failover solution
- Support for high availability architectures

1.2.11 Community Virtual Appliance Marketplace

- Marketplace with an online catalog where individuals and organizations can quickly distribute and deploy virtual appliances ready-to-run on OpenNebula cloud environments
- Marketplace is fully integrated with OpenNebula so any user of an OpenNebula cloud can find and deploy virtual appliances in a single click through familiar tools like the SunStone GUI or the OpenNebula CLI
- Support for importing OVAs processed by the AppMarket Worker. Read more here.

1.2.12 Management of Multi-tier Applications

- Automatic execution of multi-tiered applications with complete functionality for the management of groups of virtual machines as a single entity: list, delete, scale up, scale down, shutdown... and the management of Service Templates: create, show, delete, instantiate...
- Automatic deployment and undeployment of Virtual Machines according to their dependencies in the Service Template
- Provide configurable services from a catalog and self-service portal
- Enable tight, efficient administrative control
- Complete integration with the OpenNebula's User Security Management system
- Computing resources can be tracked and limited using OpenNebula's Resource Quota Management
- Automatic scaling of multi-tiered applications according to performance metrics and time schedule
- Dynamic information sharing where information can be passed across nodes in the service
- Network configuration can be defined for a service template
- OpenNebula Flow has been integrated in the Cloud and VDC Admin Sunstone views, so users can instantiate new services and monitor groups of Virtual Machines

1.2.13 Gain Insight into Cloud Applications

- OneGate allows Virtual Machine guests to push monitoring information to OpenNebula
- With a security token the VMs can call back home and report guest and/or application status in a simple way, that can be easily queried through OpenNebula interfaces (Sunstone, CLI or API).
- Users and administrators can use it to gather metrics, detect problems in their applications, and trigger OneFlow auto-scaling rules

1.2.14 Hybrid Cloud Computing and Cloud Bursting

- Extension of the local private infrastructure with resources from remote clouds
- Support for Amazon EC2 with most of the EC2 features like tags, security groups or VPC; and simultaneous access to multiple remote clouds
- Support to outsource Virtual Machines to Microsoft Azure cloud provider
- Support to outsource Virtual Machines to IBM SoftLayer cloud provider

1.2.15 Standard Cloud Interfaces and Simple Provisioning Portal for Cloud Consumers

- Transform your local infrastructure into a public cloud by exposing REST-based interfaces
- AWS EC2 API service, the de facto cloud API standard, with compatibility with EC2 ecosystem tools and client tools
- Support for simultaneously exposing multiple cloud APIs
- Provisioning portal implemented as a user Cloud View of Sunstone to allow non-IT end users to easily create, deploy and manage compute, storage and network resources

- VDCAdmin Sunstone view where VDC admins are able to create new users and manage the resources of the VDC.

1.2.16 Rich Command Line and Web Interfaces for Cloud Administrators

- Unix-like Command Line Interface to manage all resources: users, VM images, VM templates, VM instances, virtual networks, zones, VDCs, physical hosts, accounting, authentication, authorization...
- Easy-to-use Sunstone Graphical Interface providing usage graphics and statistics with cloudwatch-like functionality, remote acccess through VNC or SPICE, different system views for different roles, catalog access, multiple-zone management...
- Sunstone is easily customizable to define multiple cloud views for different user groups
- Integrated tab in Sunstone to access OpenNebula Systems (the company behind OpenNebula, formerly C12G) professional support

1.2.17 Multiple Deployment Options

- Easy to install and update with packages for most common Linux distributions
- Available in most popular Linux distributions
- Optional building from source code
- System features a small footprint, less than 10Mb
- Detailed log files with syslog support for the different components that maintain a record of significant changes
- Ability to import VMs running in hypervisors (all of them now supported, even the hybrids) that have not being launched by OpenNebula.

1.2.18 Easy Extension and Integration

- Modular and extensible architecture to fit into any existing datacenter
- Customizable drivers for the main subsystems to easily leverage existing IT infrastructure and system management products: Virtualization, Storage, Monitoring, Network, Auth and Hybrid Cloud
- New drivers can be easily written in any language
- Plugin support to easily extend SunStone Graphical Interface with additional tabs to better integrate Cloud and VM management with each site own operations and tools
- Easily customizable self-service portal for cloud consumers
- Configuration and tuning parameters to adjust behavior of the cloud management instance to the requirements of the environment and use cases
- Fully open-source technology available under Apache license
- Powerful and extensible low-level cloud API in Ruby and JAVA and XMLRPC API
- OpenNebula Add-on Catalog with components enhancing the functionality provided by OpenNebula

1.2.19 Reliability, Efficiency and Massive Scalability

- Automated testing process for functionality, scalability, performance, robustness and stability
- Technology matured through an active and engaged community
- Proven on large scale infrastructures consisting of tens of thousands of cores and VMs
- Highly scalable database back-end with support for MySQL and SQLite
- Virtualization drivers adjusted for maximum scalability
- Very efficient core developed in C++ language

(*) *Because OpenNebula leverages the functionality exposed by the underlying platform services, its functionality and performance may be affected by the limitations imposed by those services.*

- *The list of features may change on the different platform configurations*
- *Not all platform configurations exhibit a similar performance and stability*
- *The features may change to offer users more features and integration with other virtualization and cloud components*
- *The features may change due to changes in the functionality provided by underlying virtualization services*

1.3 Platform Notes

This page will show you the specific considerations at the time of using an OpenNebula cloud, according to the different supported platforms.

This is the list of the individual platform components that have been through the complete OpenNebula Quality Assurance and Certification Process.

Certified Platform Component	Version		
RedHat Enterprise Linux	6.5, 7.0		
Ubuntu Server	14.04 (LTS) , 15.04		
CentOS	6.5, 7.0		
Debian	8		
VMware	ESX 5.5/6.0 & vCenter 5.5/6.0		
XEN	3.2 & 4.2		
KVM	Support for version included in the kernel for the Linux distribution		
Amazon Web Service	Current API version		
Microsoft Azure	Current API version		
IBM SoftLayer	Current API version		

1.3.1 Front-End Platform Notes

The following applies to all Front-Ends:

- xmlrpc tuning parameters (MAX_CONN, MAX_CONN_BACKLOG, KEEPALIVE_TIMEOUT, KEEPALIVE_MAX_CONN and TIMEOUT) are only available with packages distributed by us as they are compiled with a newer xmlrpc-c library.
- for **cloud bursting**, a newer nokogiri gem than the one packed by current distros is required. If you are planning to use cloud bursting, you need to install nokogiri >= 1.4.4 prior to run `install_gems`

```
# sudo gem install nokogiri -v 1.4.4
```

- older ruby versions are not supported for **cloud bursting** (precisely for Microsoft Azure and IBM SoftLayer) and the Sunstone commercial support integration. For those supported distros with ruby versions <= 1.9.3 (like Centos 6.x) please update the ruby installation or use rvm to run a newer (>= 1.9.3) version (remember to run `install_gems` after the ruby upgrade is done to reinstall all gems)

1.3.2 CentOS Platform Notes

CentOS 6.5 Usage Platform Notes

Because home directory of oneadmin is located in `/var`, it violates SELinux default policy. So in ssh passwordless configuration you should disable SELinux by setting `SELINUX=disabled` in `/etc/selinux/config`.

CentOS 7.0 Platform Notes

This distribution lacks some packaged ruby libraries. This makes some components unusable until they are installed. In the frontend, just after package installation these command should be executed as root to install extra dependencies:

```
# /usr/share/one/install_gems
```

When using Apache to serve Sunstone, it is required that you disable or comment the `PrivateTMP=yes` directive in `/usr/lib/systemd/system/httpd.service`.

There is an automatic job that removes all data from `/var/tmp/`, in order to disable this, please edit the `/usr/lib/tmpfiles.d/tmp.conf` and re ove the line that removes `/var/tmp`.

1.3.3 RedHat 6.5 & 7.0 Platform Notes

In order to install ruby dependencies, the Server Optional channel needs to be enabled. Please refer to RedHat documentation to enable the channel.

Alternatively, use CentOS 6.5 or 7 repositories to install ruby dependencies.

1.3.4 Nodes Platform Notes

1.3.5 ESX 5.5 as VMware Node

- to accomplish disk hotplugging and nic hotplugging (ignore the first bullet for the latter)
  - disks need to be attached through SCSI, so their images should have a DEV_PREFIX="sd"
  - VM template that will permit SCSI disk attaches afterwards needs to have an explicitly defined SCSI controller:

```
RAW=[TYPE = "vmware",
     DATA = "<devices><controller type='scsi' index='0' model='lsilogic'/></devices>"]
```

- to use SCSI disk based VMs, it is usually a good idea to explicitly declare the PCI bridges. This can be accomplished with the following added to the VM template:

```
FEATURES=[PCIBRIDGE="1"]
```

- to accomplish hot migration (through vMotion)

  - VM needs to have all network card model with model "E1000"

1.3.6 CentOS 6.5 as KVM Node

- to accomplish disk hotplugging:
  - disks need to be attached through SCSI, so their images should have a DEV_PREFIX="sd"
  - VM template that will permit SCSI disk attaches afterwards needs to have an explicitly defined SCSI controller:

```
RAW=[TYPE = "kvm",
     DATA = "<devices><controller type='scsi' index='0' model='virtio-scsi'></controller></devices>"]
```

- due to libvirt version < = 0.10.2, there is a bug in libvrit/qemu attac/detach nic functionality that prevents the reuse of net IDs. This means that after a successful attach/detach NIC, a new attach will fail.

1.3.7 Unsupported Platforms Notes

Installing on ArchLinux

OpenNebula is available at the Arch User Repository (AUR), please check the opennebula package page.

Installing on Gentoo

There is an ebuild contributed by Thomas Stein in the following repository:

https://github.com/himbeere/opennebula

Still, if you want to compile it manually you need to install the xmlrpc-c package with threads support, as:

```
USE="threads" emerge xmlrpc-c
```

1.4 Compatibility Guide

This guide is aimed at OpenNebula 4.12.x users and administrators who want to upgrade to the latest version. The following sections summarize the new features and usage changes that should be taken into account, or prone to cause confusion. You can check the upgrade process in the following *guide*

Visit the *Features list* and the Release Notes for a comprehensive list of what's new in OpenNebula 4.14.

1.4.1 OpenNebula Administrators and Users

System Datastores

For OpenNebula < 4.14 access control to System Datastores is not checked by the scheduler or oned. In OpenNebula 4.12, the scheduler enforces access rights for hosts, but once a host is selected the scheduler deploys the VM in the highest ranked System DS compatible with the that host. This behavior has been changed in 4.14 and now USE rights on the System DS is required to deploy a VM in it.

You may need to update the permissions on your System DS in order to make them accessible to the users/groups. The pre-4.14 behavior can be emulated by granting USE right to OTHERS.

Virtual Data Centers

Check the previous issue about System DS; you may need to add one or more System DS to your VDCs.

Virtual Machine Recovery Process

The recover process of VMs has been redesigned in 4.14. When a VM is in `fail` state the admin has 3 choices:

- recover with success, to move on to the next state.
- recover with retry, to retry the last driver operation.
- (*only for failed migrations*) recover with failure, to cancel the migration and go back to the previous host.

As a result the previous `FAILED` state, that was final and un-recoverable, has been removed from the VM life-cycle. New states for each point of failure have been added instead.

The `onevm boot` command is deprecated. To recover a boot failure, use `onevm recover --retry` instead. Read the Virtual Machine Failures guide for more information.

Any application checking for that `FAILED` state need to be upgraded for the new LCM states:

- BOOT_FAILURE
- BOOT_MIGRATE_FAILURE
- PROLOG_MIGRATE_FAILURE
- PROLOG_FAILURE
- EPILOG_FAILURE
- EPILOG_STOP_FAILURE
- EPILOG_UNDEPLOY_FAILURE
- PROLOG_MIGRATE_POWEROFF_FAILURE
- PROLOG_MIGRATE_SUSPEND_FAILURE
- BOOT_UNDEPLOY_FAILURE
- BOOT_STOPPED_FAILURE
- PROLOG_RESUME_FAILURE
- PROLOG_UNDEPLOY_FAILURE

Virtual Machine Hooks

Hooks on `FAILED` states are no longer needed; any automatic recovery action needs to be hooked on the new `LCM_STATES`.

Default Authentication Driver

You can now specify `DEFAULT_AUTH` in `oned.conf`, meaning that it is not necessary any more to create a link called `default` pointing to the desired default authentication driver, for example `ldap`, in `/var/lib/one/remotes/auth` and it's not necessary any more to add `default` to the `authn` parameter in the `AUTH_MAD` section. However, for backwards compatibility it will continue to work, but we strongly recommend to specify directly the default auth in `oned.conf`, for example: `DEFAULT_AUTH = "ldap"`.

Virtual Machine Management

Disks and NIC attach/detach actions are now available for VMs in the `POWEROFF` state. They were previously restricted to VMs in `RUNNING` only.

Previous versions had a "save VM" functionality available through the Cloud View. This action has been redone to improve it, and make it available from other interfaces. Read more about it in the Managing Virtual Machines guide.

There are 3 `disk-snapshot` actions in **4.14**. These disk-snapshots are not related to the **4.12** action. While in 4.12 a disk-snapshot was a new image saved in the datastore, in 4.14 disk-snaphosts are similar to system snapshots. A disk has many different snapshots, and the user can revert a single disk to a previous state at any time. See the Disk Snapshots guide for more info.

- `disk-snapshot-create <vmid> <diskid> <tag>`: Creates a new snapshot of the specified disk.
- `disk-snapshot-revert <vmid> <diskid> <snapshot_id>`: Reverts to the specified snapshot. The snapshots are immutable, therefore the user can revert to the same snapshot one and again, the disk will return always to the state of the snapshot at the time it was taken.
- `disk-snapshot-delete <vmid> <diskid> <snapshot_id>`: Deletes a snapshot if it has no children and is not active.

The 4.12 `onevm disk-snapshot` action has now been renamed to `onevm disk-saveas`.

- `onevm disk-snapshot` (deferred), can now be accomplished by running `onevm poweroff` and once it's in that state, any disk can be saved by doing a new operation called `onevm disk-saveas`. Note that now you can directly run `onevm shutdown` on a machine that is in `POWEROFF` state (i.e. you don't need to resume the VM).
- `onevm disk-snapshot --live` is now called `onevm disk-saveas`

Sunstone

- The `marketplace_url` param in sunstone-server.conf should not include the /appliance path since it will be automatically included in order to support proxy configurations

1.4.2 Developers and Integrators

VM History Actions

The accounting records are individual Virtual Machine history records. A new record is created when a VM is stopped, suspended, migrated, etc. Starting in 4.14 a new record is also created when the Virtual Machine has a disk/nic attached or detached. Since the history record contains a copy of the Virtual Machine contents, this helps developers to keep track of the changes made to the disks and network interfaces of a Virtual Machine.

Virtual Machine Monitor Probes

- Monitor probes return two additional attributes (IMPORT_TEMPLATE and VM_NAME) for each found VM, to aid the import workflow.
- When the monitor probe returns state 'e' for a Virtual Machine now it is moved to the `UNKNOWN` state. In previous versions VMs went to the `FAILED` state, now removed.

Datastore Drivers

- There are 3 new Datastore Driver actions. The interface is documented in the Storage Driver guide. The end-user functionality is documented in the Images guide.
  - `snap_revert`: Overwrite the current image state with a snapshot. This operation discards any unsaved data in the current image state.
  - `snap_flatten`: Reverts the current image state to a snapshot and removes all the snapshots.
  - `snap_delete`: Deletes a snapshot.

Transfer Manager

- There are 3 new TM actions. The interface is documented in the Storage Driver guide. The end-user functionality is documented in the Virtual Machines guide.
  - `snap_create`: Handles the creation of a new disk-snapshot.
  - `snap_revert`: Overwrite the current disk state with a disk-snapshot.
  - `snap_delete`: Deletes a snapshot.
- The `mvds` now only manages saving persistent images back to the system datastore. For shared system datastores it will be a simple `exit 0`. In previous OpenNebula versions this script also served the purpose saving disk marked withed `SAVEAS` at the end of the VM lifecycle (what used to be called a deferred disk-snapshot). Since this action is no longer possible (has been replaced with `onevm disk-saveas` – see above) the `mvds` action has been largely simplified.
- The `cpds` action now accepts a `snap_id` argument. This is documented in the Storage Driver guide.

XML-RPC API

This section lists all the changes in the API. Visit the complete reference for more information.

- New API calls:
  - `one.vm.disksnapshotcreate`
  - `one.vm.disksnapshotrevert`
  - `one.vm.disksnapshotdelete`
  - `one.vm.disksaveas`
  - `one.image.snapshotdelete`
  - `one.image.snapshotrevert`
  - `one.image.snapshotflatten`
  - `one.document.lock`: New method to lock the document at the API level. The lock automatically expires after 2 minutes.
  - `one.document.unlock`: New method to unlock the document at the API level.
- Deleted API methods:
  - `one.vm.saveasdisk`
- Changed api calls:

  - `one.vm.recover` now takes an integer as argument: 0 for failure, 1 for success and 2 for retries. Applications using the pre-4.14 interface may work because of the casting of the boolean recovery operation to the new integer value. However, given the extended functionality of the new recover implementation it is recommended to review the logic of any application using this API call.
  - `one.vm.action`: The action string "boot" is not available anymore.
  - `one.template.info`: New parameter, "extended", to process the template and include extended information such as the SIZE for each DISK.

Sunstone

- The Sunstone code base has been refactored and existing plugins develovep for OpenNebula < 4.14 will not work and should be adapted to the new module oriented implentation.

1.5 Known Issues

1.5.1 CLI

- #3037 Different ruby versions need different time formats

1.5.2 Core & System

- #3020 OpenNebula should check the available space in the frontend before doing an undeploy
- #2880 Unicode chars in VM name are truncated
- #2502 deleting image in locked state leaves current operation in progress and files not cleaned
- #3139 Drivers do not work with rbenv
- #3888 An image can be used twice by a VM, but this breaks the used/ready logic

1.5.3 Drivers - Network

- #3093 Review the Open vSwitch flows
- #2961 review nic attach with 802.1Q
- #3250 WHITE_PORTS_TCP Network Filtering with Open vSwitch does not work

1.5.4 Drivers - Storage

- #1573 If an image download fails, the file is never deleted from the datastore

1.5.5 Drivers - VM

- #3590 Delete operation leaves a poweroff instance registered in vCenter
- #3750 Xen (live) migration/suspend/resume is incompatible with disk/nic attach/detach

1.5.6 OneFlow

- #3134 Service Templates with dynamic networks cannot be instantiated from the CLI, unless the a template file with the required attributes is merged
- #3797 oneflow and oneflow-template ignore the no_proxy environment variable

1.5.7 Scheduler

- #1811 If more than one scheduled actions fit in a scheduler cycle, the behaviour is unexpected

1.5.8 Sunstone

- #2292 sunstone novnc send ctrl-alt-del not working in Firefox
- #1877 if syslog enabled disable the logs tab in the VM detailed view
- #3796 sunstone ignores the no_proxy environment variable

1.6 Acknowledgements

The OpenNebula project would like to thank the community members and users who have contributed to this software release by being active with the discussions, answering user questions, or providing patches for bugfixes, features and documentation.

Disk snapshots with Ceph backend was funded by Unity in the context of the Fund a Feature Program.

Qcow2 snapshots implementation was funded by BIT.nl in the context of the Fund a Feature Program.

GPU devices support was funded by SURFsara in the context of the Fund a Feature Program.

Flexible network attributes definition in contextualization was funded by Université Catholique de Louvain in the context of the Fund a Feature Program.

1.7 Upgrading from OpenNebula 4.12.x

This guide describes the installation procedure for systems that are already running a 4.12.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the *Compatibility Guide* and Release Notes to know what is new in OpenNebula 4.14.

1.7.1 Upgrading a Federation

If you have two or more 4.12 OpenNebulas working as a Federation, you can upgrade each one independently. Zones with 4.12 and 4.14 OpenNebulas can be part of the same federation, since the shared portion of the database is compatible.

The only compatibility issue is in the Sunstone web interface. If your users access different Zones from a unique Sunstone server, you will need to upgrade all Zones to 4.14, or enable a local Sunstone server for each Zone to ensure that a 4.12 OpenNebula is only accessed through a 4.12 Sunstone. Read the federation architecture documentation for more details.

The rest of the guide applies to both a master or slave Zone. You don't need to stop the federation or the MySQL replication to follow this guide.

1.7.2 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

> **Warning:** In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: OneFlow, EC2, and Sunstone. Use preferably the system tools, like *systemctl* or *service* as *root* in order to stop the services.

1.7.3 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

```
# cp -r /etc/one /etc/one.$(date +'%Y-%m-%d')
```

1.7.4 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.12 and 4.14 versions.

1.7.5 Configuration Files Upgrade

If you haven't modified any configuration files, the package managers will replace the configuration files with their newer versions and no manual intervention is required.

If you have customized **any** configuration files under `/etc/one` we recommend you to follow these steps regardless of the platform/linux distribution.

1. Backup `/etc/one` (already performed)
2. Install the new packages (already performed)
3. Compare the old and new configuration files: `diff -ur /etc/one.YYYY-MM-DD /etc/one`. Or you can use graphical diff-tools like `meld` to compare both directories, which are very useful in this step.
4. Edit the **new** files and port all the customizations from the previous version.
5. You should **never** overwrite the configuration files with older versions.

1.7.6 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.11.80 : OpenNebula 4.12.1 daemon bootstrap

MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file


>>> Running migrators for shared tables
Database already uses version 4.11.80

>>> Running migrators for local tables
  > Running migrator /usr/lib/one/ruby/onedb/local/4.11.80_to_4.13.80.rb
************************************************************
*  WARNING  WARNING WARNING WARNING WARNING WARNING WARNING  *
************************************************************

OpenNebula 4.13.80 improves the management of FAILED VMs
Please remove (onevm delete) any FAILED VM before continuing.

************************************************************
*  WARNING  WARNING WARNING WARNING WARNING WARNING WARNING  *
************************************************************


The scheduler (and oned) has been update to enforce access
rights on system datastores. This new version also checks that
the user can access the System DS.
This *may require* to update system DS rights of your cloud

Do you want to proceed ? [y/N]y
  > Done in 41.93s
```

```
Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb command.

Total time: 41.93s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.7.7 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.12 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.7.8 Reload Start Scripts in CentOS 7

In order for the system to re-read the configuration files you should issue the following command after the installation of the new packages:

```
# systemctl daemon-reload
```

1.7.9 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.7.10 Default Auth

If you are using LDAP as default auth driver, you will need to update `/etc/one/oned.conf` and set the new `DEFAULT_AUTH` variable:

```
DEFAULT_AUTH = "ldap"
```

1.7.11 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.7.12 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.7.13 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.8 Upgrading from OpenNebula 4.10.x

This guide describes the installation procedure for systems that are already running a 4.10.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

1.8.1 Upgrading a Federation

If you have two or more 4.10.x OpenNebulas working as a Federation, you need to upgrade all of them. The upgrade does not have to be simultaneous, the slaves can be kept running while the master is upgraded.

The steps to follow are:

- 1. Stop the MySQL replication in all the slaves
- 2. Upgrade the **master** OpenNebula
- 3. Upgrade each **slave**
- 4. Resume the replication

During the time between steps 1 and 4 the slave OpenNebulas can be running, and users can keep accessing them if each zone has a local Sunstone instance. There is however an important limitation to note: all the shared database tables will not be updated in the slaves zones. This means that new user accounts, password changes, new ACL rules, etc. will not have any effect in the slaves. Read the federation architecture documentation for more details.

It is recommended to upgrade all the slave zones as soon as possible.

To perform the first step, pause the replication in each **slave MySQL**:

```
mysql> STOP SLAVE;

mysql> SHOW SLAVE STATUS\G

 Slave_IO_Running: No
Slave_SQL_Running: No
```

Then follow this guide for the **master zone**. After the master has been updated to 4.14, upgrade each **slave zone** following this same guide.

1.8.2 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: OneFlow, EC2, and Sunstone. Use preferably the system tools, like *systemctl* or *service* as *root* in order to stop the services.

1.8.3 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

```
# cp -r /etc/one /etc/one.$(date +'%Y-%m-%d')
```

1.8.4 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.10 and 4.14 versions.

1.8.5 Configuration Files Upgrade

If you haven't modified any configuration files, the package managers will replace the configuration files with their newer versions and no manual intervention is required.

If you have customized **any** configuration files under `/etc/one` we recommend you to follow these steps regardless of the platform/linux distribution.

1. Backup `/etc/one` (already performed)
2. Install the new packages (already performed)
3. Compare the old and new configuration files: `diff -ur /etc/one.YYYY-MM-DD /etc/one`. Or you can use graphical diff-tools like `meld` to compare both directories, which are very useful in this step.
4. Edit the **new** files and port all the customizations from the previous version.
5. You should **never** overwrite the configuration files with older versions.

1.8.6 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Warning: For environments in a Federation: Before upgrading the **master**, make sure that all the slaves have the MySQL replication paused.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap

>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.8.7 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.10 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.8.8 Resume the Federation

This section applies only to environments working in a Federation.

For the **master zone**: This step is not necessary.

For a **slave zone**: The MySQL replication must be resumed now.

- First, add a new table, `vdc_pool`, to the replication configuration.

Warning: Do not copy the server-id from this example, each slave should already have a unique ID.

```
# vi /etc/my.cnf
[mysqld]
server-id           = 100
replicate-do-table  = opennebula.user_pool
replicate-do-table  = opennebula.group_pool
replicate-do-table  = opennebula.vdc_pool
replicate-do-table  = opennebula.zone_pool
replicate-do-table  = opennebula.db_versioning
replicate-do-table  = opennebula.acl

# service mysqld restart
```

- Start the **slave MySQL** process and check its status. It may take a while to copy and apply all the pending commands.

```
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS\G
```

The `SHOW SLAVE STATUS` output will provide detailed information, but to confirm that the slave is connected to the master MySQL, take a look at these columns:

```
   Slave_IO_State: Waiting for master to send event
 Slave_IO_Running: Yes
Slave_SQL_Running: Yes
```

1.8.9 Reload Start Scripts in CentOS 7

In order for the system to re-read the configuration files you should issue the following command after the installation of the new packages:

```
# systemctl daemon-reload
```

1.8.10 Enable Start Scripts in CentOS 7

CentOS 7 packages now come with systemd scripts instead of the old systemV ones. You will need to enable the services again so they are started on system boot. The names of the services are the same as the previous one. For example, to enable `opennebula`, `opennebula-sunstone`, `opennebula-flow` and `opennebula-gate` you can issue these commands:

```
# systemctl enable opennebula
# systemctl enable opennebula-sunstone
# systemctl enable opennebula-flow
# systemctl enable opennebula-gate
```

1.8.11 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.8.12 Default Auth

If you are using LDAP as default auth driver, you will need to update `/etc/one/oned.conf` and set the new `DEFAULT_AUTH` variable:

```
DEFAULT_AUTH = "ldap"
```

1.8.13 vCenter Password

Note: This step only applies if you are upgrading from OpenNebula **4.10.0**. If you are already using 4.10.1 or 4.10.2 you can skip this step.

If you already have a host with vCenter drivers you need to update the password as version >4.10.0 expects it to be encrypted. To do so, proceed to Sunstone -> Infrastructure -> Hosts, click on the vCenter host(s) and change the value in `VCENTER_PASSWORD` field. It will be automatically encrypted.

1.8.14 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 4.14.

1.8.15 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.8.16 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.

- Copy back the backup of /etc/one you did to restore your configuration.

1.8.17 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.9 Upgrading from OpenNebula 4.8.x

This guide describes the installation procedure for systems that are already running a 4.8.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

1.9.1 Upgrading a Federation

If you have two or more 4.8 OpenNebulas working as a Federation, you need to upgrade all of them. The upgrade does not have to be simultaneous, the slaves can be kept running while the master is upgraded.

The steps to follow are:

- 1. Stop the MySQL replication in all the slaves
- 2. Upgrade the **master** OpenNebula
- 3. Upgrade each **slave**
- 4. Resume the replication

During the time between steps 1 and 4 the slave OpenNebulas can be running, and users can keep accessing them if each zone has a local Sunstone instance. There is however an important limitation to note: all the shared database tables will not be updated in the slaves zones. This means that new user accounts, password changes, new ACL rules, etc. will not have any effect in the slaves. Read the federation architecture documentation for more details.

It is recommended to upgrade all the slave zones as soon as possible.

To perform the first step, pause the replication in each **slave MySQL**:

```
mysql> STOP SLAVE;

mysql> SHOW SLAVE STATUS\G

 Slave_IO_Running: No
Slave_SQL_Running: No
```

Then follow this guide for the **master zone**. After the master has been updated to 4.14, upgrade each **slave zone** following this same guide.

1.9.2 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ one stop
```

1.9.3 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

```
# cp -r /etc/one /etc/one.YYYY-MM-DD
```

Note: Substitute `YYYY-MM-DD` with the date.

1.9.4 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.8 and 4.14 versions.

1.9.5 Configuration Files Upgrade

If you haven't modified any configuration files, the package managers will replace the configuration files with their newer versions and no manual intervention is required.

If you have customized **any** configuration files under `/etc/one` we recommend you to follow these steps regardless of the platform/linux distribution.

1. Backup `/etc/one` (already performed)
2. Install the new packages (already performed)
3. Compare the old and new configuration files: `diff -ur /etc/one.YYYY-MM-DD /etc/one`. Or you can use graphical diff-tools like `meld` to compare both directories, which are very useful in this step.
4. Edit the **new** files and port all the customizations from the previous version.
5. You should **never** overwrite the configuration files with older versions.

1.9.6 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Warning: For environments in a Federation: Before upgrading the **master**, make sure that all the slaves have the MySQL replication paused.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap

>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.9.7 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.8 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.9.8 Resume the Federation

This section applies only to environments working in a Federation.

For the **master zone**: This step is not necessary.

For a **slave zone**: The MySQL replication must be resumed now.

- First, add a new table, `vdc_pool`, to the replication configuration.

Warning: Do not copy the server-id from this example, each slave should already have a unique ID.

```
# vi /etc/my.cnf
[mysqld]
server-id           = 100
replicate-do-table  = opennebula.user_pool
replicate-do-table  = opennebula.group_pool
replicate-do-table  = opennebula.vdc_pool
replicate-do-table  = opennebula.zone_pool
replicate-do-table  = opennebula.db_versioning
replicate-do-table  = opennebula.acl

# service mysqld restart
```

- Start the **slave MySQL** process and check its status. It may take a while to copy and apply all the pending commands.

```
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS\G
```

The `SHOW SLAVE STATUS` output will provide detailed information, but to confirm that the slave is connected to the master MySQL, take a look at these columns:

```
   Slave_IO_State: Waiting for master to send event
 Slave_IO_Running: Yes
Slave_SQL_Running: Yes
```

1.9.9 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.9.10 Default Auth

If you are using LDAP as default auth driver, you will need to update `/etc/one/oned.conf` and set the new `DEFAULT_AUTH` variable:

```
DEFAULT_AUTH = "ldap"
```

1.9.11 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 4.14.

1.9.12 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.9.13 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.9.14 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.10 Upgrading from OpenNebula 4.6.x

This guide describes the installation procedure for systems that are already running a 4.6.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.8, 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

1.10.1 Upgrading a Federation

If you have two or more 4.6 OpenNebulas working as a Federation, you need to upgrade all of them. The upgrade does not have to be simultaneous, the slaves can be kept running while the master is upgraded.

The steps to follow are:

- 1. Stop the MySQL replication in all the slaves
- 2. Upgrade the **master** OpenNebula
- 3. Upgrade each **slave**
- 4. Resume the replication

During the time between steps 1 and 4 the slave OpenNebulas can be running, and users can keep accessing them if each zone has a local Sunstone instance. There is however an important limitation to note: all the shared database tables will not be updated in the slaves zones. This means that new user accounts, password changes, new ACL rules, etc. will not have any effect in the slaves. Read the federation architecture documentation for more details.

It is recommended to upgrade all the slave zones as soon as possible.

To perform the first step, pause the replication in each **slave MySQL**:

```
mysql> STOP SLAVE;

mysql> SHOW SLAVE STATUS\G

 Slave_IO_Running: No
Slave_SQL_Running: No
```

Then follow this guide for the **master zone**. After the master has been updated to 4.14, upgrade each **slave zone** following this same guide.

1.10.2 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ occi-server stop
$ one stop
```

1.10.3 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

```
# cp -r /etc/one /etc/one.YYYY-MM-DD
```

Note: Substitute `YYYY-MM-DD` with the date.

1.10.4 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.6 and 4.14 versions.

1.10.5 Configuration Files Upgrade

If you haven't modified any configuration files, the package managers will replace the configuration files with their newer versions and no manual intervention is required.

If you have customized **any** configuration files under `/etc/one` we recommend you to follow these steps regardless of the platform/linux distribution.

1. Backup `/etc/one` (already performed)
2. Install the new packages (already performed)
3. Compare the old and new configuration files: `diff -ur /etc/one.YYYY-MM-DD /etc/one`. Or you can use graphical diff-tools like `meld` to compare both directories, which are very useful in this step.
4. Edit the **new** files and port all the customizations from the previous version.

5. You should **never** overwrite the configuration files with older versions.

1.10.6 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Warning: For environments in a Federation: Before upgrading the **master**, make sure that all the slaves have the MySQL replication paused.

Note: If you have a MAC_PREFIX in oned.conf different than the default `02:00`, open `/usr/lib/one/ruby/onedb/local/4.5.80_to_4.7.80.rb` and change the value of the `ONEDCONF_MAC_PREFIX` constant.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap

>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
```

```
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.10.7 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.6 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.10.8 Resume the Federation

This section applies only to environments working in a Federation.

For the **master zone**: This step is not necessary.

For a **slave zone**: The MySQL replication must be resumed now.

- First, add a new table, `vdc_pool`, to the replication configuration.

Warning: Do not copy the server-id from this example, each slave should already have a unique ID.

```
# vi /etc/my.cnf
[mysqld]
server-id           = 100
replicate-do-table  = opennebula.user_pool
replicate-do-table  = opennebula.group_pool
replicate-do-table  = opennebula.vdc_pool
replicate-do-table  = opennebula.zone_pool
replicate-do-table  = opennebula.db_versioning
replicate-do-table  = opennebula.acl

# service mysqld restart
```

- Start the **slave MySQL** process and check its status. It may take a while to copy and apply all the pending commands.

```
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS\G
```

The `SHOW SLAVE STATUS` output will provide detailed information, but to confirm that the slave is connected to the master MySQL, take a look at these columns:

```
   Slave_IO_State: Waiting for master to send event
 Slave_IO_Running: Yes
Slave_SQL_Running: Yes
```

1.10.9 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.10.10 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 4.14.

1.10.11 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.10.12 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.10.13 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.11 Upgrading from OpenNebula 4.4.x

This guide describes the installation procedure for systems that are already running a 4.4.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.6, 4.8, 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

1.11.1 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

> **Warning:** In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ occi-server stop
$ one stop
```

1.11.2 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

1.11.3 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.4 and 4.14 versions.

1.11.4 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Note: If you have a MAC_PREFIX in oned.conf different than the default `02:00`, open `/usr/lib/one/ruby/onedb/local/4.5.80_to_4.7.80.rb` and change the value of the `ONEDCONF_MAC_PREFIX` constant.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap

>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.11.5 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.4 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.11.6 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.11.7 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

1.11.8 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.11.9 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.11.10 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.12 Upgrading from OpenNebula 4.2

This guide describes the installation procedure for systems that are already running a 4.2 OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.4, 4.6, 4.8, 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

Warning: With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

Warning: Two drivers available in 4.0 are now discontinued: **ganglia** and **iscsi**.
- **iscsi** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.
- **ganglia** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.

1.12.1 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ occi-server stop
$ one stop
```

1.12.2 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

1.12.3 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.2 and 4.14 versions.

1.12.4 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Note: If you have a MAC_PREFIX in oned.conf different than the default `02:00`, open `/usr/lib/one/ruby/onedb/local/4.5.80_to_4.7.80.rb` and change the value of the `ONEDCONF_MAC_PREFIX` constant.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap
```

```
>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

If you receive the message "ATTENTION: manual intervention required", read the section *Manual Intervention Required* below.

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.12.5 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.2 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.12.6 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.12.7 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

1.12.8 Setting new System DS

With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

You will need to have at least one system DS in each cluster. If you don't already, create new system DS with the same definition as the system DS 0 (TM_MAD driver). Depending on your setup this may or may not require additional configuration on the hosts.

You may also try to recover saved VMs (stop, suspend, undeploy) following the steps described in this thread of the users mailing list.

1.12.9 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.12.10 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.12.11 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.12.12 Manual Intervention Required

If you have a datastore configured to use a tm driver not included in the OpenNebula distribution, the onedb upgrade command will show you this message:

```
ATTENTION: manual intervention required

The Datastore <id> <name> is using the
custom TM MAD '<tm_mad>'. You will need to define new
configuration parameters in oned.conf for this driver, see
http://opennebula.org/documentation:rel4.4:upgrade
```

Since OpenNebula 4.4, each tm_mad driver has a TM_MAD_CONF section in oned.conf. If you developed the driver, it should be fairly easy to define the required information looking at the existing ones:

```
# The  configuration for each driver is defined in TM_MAD_CONF. These
# values are used when creating a new datastore and should not be modified
# since they define the datastore behaviour.
#   name      : name of the transfer driver, listed in the -d option of the
#               TM_MAD section
#   ln_target : determines how the persistent images will be cloned when
#               a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   clone_target : determines how the non persistent images will be
#                  cloned when a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   shared : determines if the storage holding the system datastore is shared
#            among the different hosts or not. Valid values: "yes" or "no"

TM_MAD_CONF = [
    name        = "lvm",
    ln_target   = "NONE",
    clone_target= "SELF",
    shared      = "yes"
]
```

1.13 Upgrading from OpenNebula 4.0.x

This guide describes the installation procedure for systems that are already running a 4.0.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.2, 4.4, 4.6, 4.8, 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

Warning: With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

Warning: Two drivers available in 4.0 are now discontinued: **ganglia** and **iscsi**.
- **iscsi** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.
- **ganglia** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.

Warning: There are combinations of **VMware storage** no longer supported (see the VMFS Datastore guide for the supported configurations).
If you want to upgrade and you are using SSH, NFS or VMFS without SSH-mode, you will need to manually migrate your images to a newly created VMFS with SSH-mode datastore. To do so implies powering off all the VMs with images in any of the deprecated datastores, upgrade OpenNebula, create a VMFS datastore and then manually register the images from those deprecated datastores into the new one. Let us know if you have doubts or problems with this process.

1.13.1 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ occi-server stop
$ one stop
```

1.13.2 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

1.13.3 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 4.0 and 4.14 versions.

1.13.4 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Note: If you have a MAC_PREFIX in oned.conf different than the default `02:00`, open `/usr/lib/one/ruby/onedb/local/4.5.80_to_4.7.80.rb` and change the value of the `ONEDCONF_MAC_PREFIX` constant.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 4.4.0 : OpenNebula 4.4.0 daemon bootstrap
Local tables  4.4.0 : OpenNebula 4.4.0 daemon bootstrap

>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.75s

Database migrated from 4.4.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80
Total time: 0.77s
```

If you receive the message "ATTENTION: manual intervention required", read the section *Manual Intervention Required* below.

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done
```

```
  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.13.5 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.0 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.13.6 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.13.7 Setting new System DS

With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

You will need to have at least one system DS in each cluster. If you don't already, create new system DS with the same definition as the system DS 0 (TM_MAD driver). Depending on your setup this may or may not require additional configuration on the hosts.

You may also try to recover saved VMs (stop, suspend, undeploy) following the steps described in this thread of the users mailing list.

1.13.8 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

1.13.9 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.13.10 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.13.11 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```

1.13.12 Manual Intervention Required

If you have a datastore configured to use a tm driver not included in the OpenNebula distribution, the onedb upgrade command will show you this message:

```
ATTENTION: manual intervention required

The Datastore <id> <name> is using the
custom TM MAD '<tm_mad>'. You will need to define new
configuration parameters in oned.conf for this driver, see
http://opennebula.org/documentation:rel4.4:upgrade
```

Since OpenNebula 4.4, each tm_mad driver has a TM_MAD_CONF section in oned.conf. If you developed the driver, it should be fairly easy to define the required information looking at the existing ones:

```
# The  configuration for each driver is defined in TM_MAD_CONF. These
# values are used when creating a new datastore and should not be modified
# since they define the datastore behaviour.
#   name      : name of the transfer driver, listed in the -d option of the
#               TM_MAD section
#   ln_target : determines how the persistent images will be cloned when
#               a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   clone_target : determines how the non persistent images will be
#                  cloned when a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   shared : determines if the storage holding the system datastore is shared
#            among the different hosts or not. Valid values: "yes" or "no"

TM_MAD_CONF = [
    name        = "lvm",
    ln_target   = "NONE",
    clone_target= "SELF",
    shared      = "yes"
]
```

1.14 Upgrading from OpenNebula 3.8.x

This guide describes the installation procedure for systems that are already running a 3.8.x OpenNebula. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.0, 4.2, 4.4, 4.6, 4.8, 4.10, 4.12 and *4.14*, and the Release Notes to know what is new in OpenNebula 4.14.

Warning: With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

Warning: Two drivers available in 3.8 are now discontinued: **ganglia** and **iscsi**.

- **iscsi** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.
- **ganglia** drivers have been moved out of the main OpenNebula distribution and are available (although not supported) as an addon.

Warning: There are combinations of **VMware storage** no longer supported (see the VMFS Datastore guide for the supported configurations).
If you want to upgrade and you are using SSH, NFS or VMFS without SSH-mode, you will need to manually migrate your images to a newly created VMFS with SSH-mode datastore. To do so implies powering off all the VMs with images in any of the deprecated datastores, upgrade OpenNebula, create a VMFS datastore and then manually register the images from those deprecated datastores into the new one. Let us know if you have doubts or problems with this process.

1.14.1 Preparation

Before proceeding, make sure you don't have any VMs in a transient state (prolog, migr, epil, save). Wait until these VMs get to a final state (runn, suspended, stopped, done). Check the Managing Virtual Machines guide for more information on the VM life-cycle.

Warning: In 4.14 the `FAILED` state dissapears. You need to delete all the VMs in this state **before** the new version is installed.

Stop OpenNebula and any other related services you may have running: EC2, OCCI, and Sunstone. As `oneadmin`, in the front-end:

```
$ sunstone-server stop
$ oneflow-server stop
$ econe-server stop
$ occi-server stop
$ one stop
```

1.14.2 Backup

Backup the configuration files located in **/etc/one**. You don't need to do a manual backup of your database, the onedb command will perform one automatically.

1.14.3 Installation

Follow the *Platform Notes* and the Installation guide, taking into account that you will already have configured the passwordless ssh access for oneadmin.

Make sure to run the `install_gems` tool, as the new OpenNebula version may have different gem requirements.

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 4.14 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the *Compatibility Guide* and the complete oned.conf reference for 3.8 and 4.14 versions.

1.14.4 Database Upgrade

The database schema and contents are incompatible between versions. The OpenNebula daemon checks the existing DB version, and will fail to start if the version found is not the one expected, with the message 'Database version mismatch'.

You can upgrade the existing DB with the 'onedb' command. You can specify any Sqlite or MySQL database. Check the onedb reference for more information.

Warning: Make sure at this point that OpenNebula is not running. If you installed from packages, the service may have been started automatically.

Note: If you have a MAC_PREFIX in oned.conf different than the default `02:00`, open `/usr/lib/one/ruby/onedb/local/4.5.80_to_4.7.80.rb` and change the value of the `ONEDCONF_MAC_PREFIX` constant.

After you install the latest OpenNebula, and fix any possible conflicts in oned.conf, you can issue the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. Some examples:

```
$ onedb upgrade -v --sqlite /var/lib/one/one.db
```

```
$ onedb upgrade -v -S localhost -u oneadmin -p oneadmin -d opennebula
```

If everything goes well, you should get an output similar to this one:

```
$ onedb upgrade -v -u oneadmin -d opennebula
MySQL Password:
Version read:
Shared tables 3.8.0 : OpenNebula 3.8.0 daemon bootstrap
Local tables  3.8.0 : OpenNebula 3.8.0 daemon bootstrap

MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file


>>> Running migrators for shared tables
  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.0_to_3.8.1.rb
  > Done in 0.36s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.1_to_3.8.2.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.2_to_3.8.3.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.3_to_3.8.4.rb
  > Done in 0.56s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.4_to_3.8.5.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.8.5_to_3.9.80.rb

ATTENTION: manual intervention required
Virtual Machine deployment files have been moved from /var/lib/one to
/var/lib/one/vms. You need to move these files manually:

    $ mv /var/lib/one/[0-9]* /var/lib/one/vms

  > Done in 1.10s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.9.80_to_3.9.90.rb

ATTENTION: manual intervention required
IM and VM MADS have been renamed in oned.conf. To keep your
existing hosts working, you need to duplicate the drivers with the
old names.

For example, for kvm you will have IM_MAD "kvm" and VM_MAD "kvm", so you
need to add IM_MAD "im_kvm" and VM_MAD "vmm_kvm"

IM_MAD = [
      name       = "kvm",
      executable = "one_im_ssh",
      arguments  = "-r 0 -t 15 kvm" ]
```

```
IM_MAD = [
      name       = "im_kvm",
      executable = "one_im_ssh",
      arguments  = "-r 0 -t 15 kvm" ]

VM_MAD = [
    name       = "kvm",
    executable = "one_vmm_exec",
    arguments  = "-t 15 -r 0 kvm",
    default    = "vmm_exec/vmm_exec_kvm.conf",
    type       = "kvm" ]

VM_MAD = [
    name       = "vmm_kvm",
    executable = "one_vmm_exec",
    arguments  = "-t 15 -r 0 kvm",
    default    = "vmm_exec/vmm_exec_kvm.conf",
    type       = "kvm" ]

  > Done in 0.41s

  > Running migrator /usr/lib/one/ruby/onedb/shared/3.9.90_to_4.0.0.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.0.0_to_4.0.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.0.1_to_4.1.80.rb
  > Done in 0.09s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.1.80_to_4.2.0.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.2.0_to_4.3.80.rb
  > Done in 0.68s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.3.80_to_4.3.85.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.3.85_to_4.3.90.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.3.90_to_4.4.0.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.0_to_4.4.1.rb
  > Done in 0.00s

  > Running migrator /usr/lib/one/ruby/onedb/shared/4.4.1_to_4.5.80.rb
  > Done in 0.39s

Database migrated from 3.8.0 to 4.5.80 (OpenNebula 4.5.80) by onedb command.

>>> Running migrators for local tables
Database already uses version 4.5.80

Total time: 3.60s
```

Now execute the following DB patch:

```
$ onedb patch -v -u oneadmin -d opennebula /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
Version read:
Shared tables 4.11.80 : OpenNebula 4.12.1 daemon bootstrap
Local tables  4.13.80 : Database migrated from 4.11.80 to 4.13.80 (OpenNebula 4.13.80) by onedb comma

  > Running patch /usr/lib/one/ruby/onedb/patches/4.14_monitoring.rb
  > Done

  > Total time: 0.05s
```

Warning: This DB upgrade is expected to take a long time to complete in large infrastructures. If you have an OpenNebula Systems support subscription, please contact them to study your case and perform the upgrade with the minimum downtime possible.

Note: Make sure you keep the backup file. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions.

1.14.5 Check DB Consistency

After the upgrade is completed, you should run the command `onedb fsck`.

First, move the 4.0 backup file created by the upgrade command to a safe place.

```
$ mv /var/lib/one/mysql_localhost_opennebula.sql /path/for/one-backups/
```

Then execute the following command:

```
$ onedb fsck -S localhost -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file

Total errors found: 0
```

1.14.6 Virtual Machine Directories

Note: Only for OpenNebula versions < 3.8.3

If you are upgrading from a version **lower than 3.8.3**, you need to move the Virtual Machine deployment files from '/var/lib/one' to '/var/lib/one/vms':

```
$ mv /var/lib/one/[0-9]* /var/lib/one/vms
```

1.14.7 Driver Names

OpenNebula default driver names have changed in the configuration file. Now the names of the vmm and im drivers are not prepended by the type of driver:

- vmm_kvm → kvm
- vmm_xen → xen

- vmm_vmware → vmware
- vmm_ec2 → ec2
- vmm_dummy → dummy
- im_kvm → kvm
- im_xen → xen
- im_vmware → vmware
- im_ec2 → ec2
- im_ganglia → ganglia
- im_dummy → dummy

To keep your existing hosts working, you need to duplicate the drivers with the old names.

For example, for kvm you will have IM_MAD `kvm` and VM_MAD `kvm`, so you need to add IM_MAD `im_kvm` and VM_MAD `vmm_kvm`

```
IM_MAD = [
      name       = "kvm",
      executable = "one_im_ssh",
      arguments  = "-r 3 -t 15 kvm" ]

IM_MAD = [
      name       = "im_kvm",
      executable = "one_im_ssh",
      arguments  = "-r 3 -t 15 kvm" ]

VM_MAD = [
    name       = "kvm",
    executable = "one_vmm_exec",
    arguments  = "-t 15 -r 0 kvm",
    default    = "vmm_exec/vmm_exec_kvm.conf",
    type       = "kvm" ]

VM_MAD = [
    name       = "vmm_kvm",
    executable = "one_vmm_exec",
    arguments  = "-t 15 -r 0 kvm",
    default    = "vmm_exec/vmm_exec_kvm.conf",
    type       = "kvm" ]
```

1.14.8 Manual Intervention Required

Note: Ignore this section if onedb didn't output the following message

If you have a datastore configured to use a tm driver not included in the OpenNebula distribution, the onedb upgrade command will show you this message:

```
ATTENTION: manual intervention required

The Datastore <id> <name> is using the
custom TM MAD '<tm_mad>'. You will need to define new
configuration parameters in oned.conf for this driver, see
http://opennebula.org/documentation:rel4.4:upgrade
```

Since OpenNebula 4.4, each tm_mad driver has a TM_MAD_CONF section in oned.conf. If you developed the driver, it should be fairly easy to define the required information looking at the existing ones:

```
# The  configuration for each driver is defined in TM_MAD_CONF. These
# values are used when creating a new datastore and should not be modified
# since they define the datastore behaviour.
#   name      : name of the transfer driver, listed in the -d option of the
#               TM_MAD section
#   ln_target : determines how the persistent images will be cloned when
#               a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   clone_target : determines how the non persistent images will be
#                  cloned when a new VM is instantiated.
#       NONE: The image will be linked and no more storage capacity will be used
#       SELF: The image will be cloned in the Images datastore
#       SYSTEM: The image will be cloned in the System datastore
#   shared : determines if the storage holding the system datastore is shared
#            among the different hosts or not. Valid values: "yes" or "no"

TM_MAD_CONF = [
    name        = "lvm",
    ln_target   = "NONE",
    clone_target= "SELF",
    shared      = "yes"
]
```

1.14.9 Update the Drivers

You should be able now to start OpenNebula as usual, running 'one start' as oneadmin. At this point, execute `onehost sync` to update the new drivers in the hosts.

Warning: Doing `onehost sync` is important. If the monitorization drivers are not updated, the hosts will behave erratically.

1.14.10 Setting new System DS

With the new multi-system DS functionality, it is now required that the system DS is also part of the cluster. If you are using System DS 0 for Hosts inside a Cluster, any VM saved (stop, suspend, undeploy) **will not be able to be resumed after the upgrade process**.

You will need to have at least one system DS in each cluster. If you don't already, create new system DS with the same definition as the system DS 0 (TM_MAD driver). Depending on your setup this may or may not require additional configuration on the hosts.

You may also try to recover saved VMs (stop, suspend, undeploy) following the steps described in this thread of the users mailing list.

1.14.11 Create the Security Group ACL Rule

There is a new kind of resource introduced in 4.12: Security Groups. If you want your existing users to be able to create their own Security Groups, create the following ACL Rule:

```
$ oneacl create "* SECGROUP/* CREATE *"
```

1.14.12 Testing

OpenNebula will continue the monitoring and management of your previous Hosts and VMs.

As a measure of caution, look for any error messages in oned.log, and check that all drivers are loaded successfully. After that, keep an eye on oned.log while you issue the onevm, onevnet, oneimage, oneuser, onehost **list** commands. Try also using the **show** subcommand for some resources.

1.14.13 Restoring the Previous Version

If for any reason you need to restore your previous OpenNebula, follow these steps:

- With OpenNebula 4.14 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 4.14, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

1.14.14 Known Issues

If the MySQL database password contains special characters, such as `@` or `#`, the onedb command will fail to connect to it.

The workaround is to temporarily change the oneadmin's password to an ASCII string. The set password statement can be used for this:

```
$ mysql -u oneadmin -p

mysql> SET PASSWORD = PASSWORD('newpass');
```