OpenNebula 5.2 Introduction and Release Notes

Release 5.2.1

OpenNebula Systems

Jan 09, 2017

CONTENTS

CHAPTER

ONE

CONCEPTS AND TERMINOLOGY

1.1 Start Here: OpenNebula Overview

Welcome to OpenNebula documentation!

OpenNebula is an open-source management platform to build IaaS private, public and hybrid clouds. Installing a cloud from scratch could be a complex process, in the sense that many components and concepts are involved. The degree of familiarity with these concepts (system administration, infrastructure planning, virtualization management...) will determine the difficulty of the installation process.

If you are new to OpenNebula you should go through this short introduction before proceeding to the deployment and administration guides.

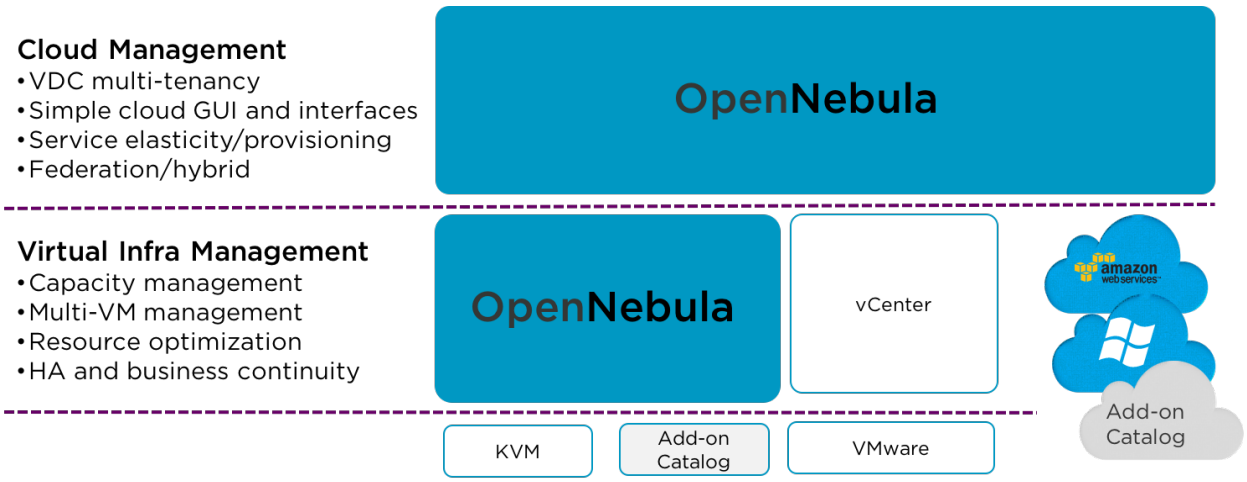
1.1.1 Step 1. Choose Your Hypervisor

The first step is to decide on the hypervisor that you will use in your cloud infrastructure. The main OpenNebula distribution provides full support for the two most widely used hypervisors, KVM and VMware (through vCenter), at different levels of functionality.

- **Virtualization and Cloud Management on KVM**. Many companies use OpenNebula to manage data center virtualization, consolidate servers, and integrate existing IT assets for computing, storage, and networking. In this deployment model, OpenNebula directly integrates with KVM and has complete control over virtual and physical resources, providing advanced features for capacity management, resource optimization, high availability and business continuity. Some of these deployments additionally use OpenNebula's **Cloud Management and Provisioning** features when they want to federate data centers, implement cloudbursting, or offer self-service portals for end users.
- **Cloud Management on VMware vCenter**. Other companies use OpenNebula to provide a multi-tenant, cloud-like provisioning layer on top of VMware vCenter. These deployments are looking for provisioning, elasticity and multi-tenancy cloud features like virtual data centers provisioning, datacenter federation or hybrid cloud computing to connect in-house infrastructures with public clouds, while the infrastructure is managed by already familiar tools for infrastructure management and operation, such as vSphere and vCenter Operations Manager.

After having installed the cloud with one hypervisor you may add another hypervisors. You can deploy heterogeneous multi-hypervisor environments managed by a single OpenNebula instance. An advantage of using OpenNebula on VMware is the strategic path to openness as companies move beyond virtualization toward a private cloud. OpenNebula can leverage existing VMware infrastructure, protecting IT investments, and at the same time gradually integrate other open-source hypervisors, therefore avoiding future vendor lock-in and strengthening the negotiating position of the company.

There are other virtualization technologies, like LXC or Xen, supported by the community. Please refer to the OpenNebula Add-ons Catalog.



1.1.2 Step 2. Design and Install the Cloud

2.1. Design the Cloud Architecture

In order to get the most out of a OpenNebula Cloud, we recommend that you create a plan with the features, performance, scalability, and high availability characteristics you want in your deployment. We have prepared **Cloud Architecture Design guides** for KVM and vCenter to help you plan an OpenNebula installation, so you can easily architect your deployment and understand the technologies involved in the management of virtualized resources and their relationship. These guides have been created from the collective information and experiences from hundreds of users and cloud client engagements. Besides main logical components and interrelationships, this guides document software products, configurations, and requirements of infrastructure platforms recommended for a smooth OpenNebula installation.

2.2. Install the Front-end

Next step is the **installation of OpenNebula in the cloud front-end**. This installation process is the same for any underlying hypervisor.

Optionally you can setup a high available cluster for OpenNebula for OpenNebula to reduce downtime of core OpenNebula services, and configure a MySQL backend as an alternative to the default Sqlite backend if you are planning a large-scale infrastructure.

2.3. Install the Virtualization hosts

Now you are ready to **add the virtualization nodes**. The OpenNebula packages bring support for KVM and vCenter nodes. In the case of vCenter, a host represents a vCenter cluster with all its ESX hosts. You can add different hypervisors to the same OpenNebula instance, or any other virtualization technology, like LXC or Xen, supported by the community. Please refer to the OpenNebula Add-ons Catalog.

1.1.3 Step 3. Set-up Infrastructure and Services

3.1. Integrate with Data Center Infrastructure

Now you should have an OpenNebula cloud up and running with at least one virtualization node. The next step is, if needed, to perform the integration of OpenNebula with your infrastructure platform and define the configuration of

its components. When using the vCenter driver, no additional integration is required because the interaction with the underlying networking, storage and compute infrastructure is performed through vCenter.

However when using KVM, in the open cloud architecture, OpenNebula directly manages the hypervisor, networking and storage platforms, and you may need additional configuration:

- **Networking setup** with 802.1Q VLANs, ebtables, Open vSwitch or VXLAN.
- **Storage setup** with filesystem datastore, LVM datastore, Ceph, Dev, or iSCSI datastore.
- **Host setup** with the configuration options for the KVM hosts, Monitoring subsystem, Virtual Machine HA or PCI Passthrough.

3.2. Configure Cloud Services

OpenNebula comes by default with an internal **user/password authentication system**. Optionally you can enable an external Authentication driver like ssh, x509, ldap or Active Directory.

Sunstone, the OpenNebula GUI, brings by default a pre-defined configuration of views. Optionally it can be customized and extended to meet your needs. You can customize the roles and views, improve security with x509 authentication and SSL or improve scalability for large deployments.

We also provide **references** with a detailed description of the different configuration files, and logging and debugging reports of the OpenNebula services.

1.1.4 Step 4. Operate your Cloud

4.1. Define a Provisioning Model

Before configuring multi-tenancy and defining the provisioning model of your cloud, we recommend you go through this introduction to the OpenNebula provisioning model. In a small installation with a few hosts, you can skip this guide and use OpenNebula without giving much thought to infrastructure partitioning and provisioning. But for medium and large deployments you will probably want to provide some level of isolation and structure.

- Regarding the **underlying infrastructure**, OpenNebula provides complete functionality for the management of the physical hosts and clusters in the cloud. A Cluster is a group of Hosts that can have associated Datastores and Virtual Networks.
- Regarding **user management**, OpenNebula features advanced multi-tenancy with powerful users and groups management, a Access Control List mechanism allowing different role management with fine grain permission granting over any resource, resource quota management to track and limit computing, storage and networking utilization, and a configurable accounting and showback systems to visualize and report resource usage data and to allow their integration with chargeback and billing platforms, or to guarantee fair share of resources among users.
- Last but not least, you can define VDCs (Virtual Data Center) as assignments of one or several user groups to a pool of physical resources. While clusters are used to group physical resources according to common characteristics such as networking topology or physical location, Virtual Data Centers (VDCs) allow to create "logical" pools of resources (which could belong to different clusters and cones) and allocate them to user groups.

4.2. Manage Virtual Resources

Now everything is ready for operation. OpenNebula provides full control to manage virtual resources.

- **Virtual machine image management** that allows to store disk images in catalogs (termed datastores), that can be then used to define VMs or shared with other users. The images can be OS installations, persistent data sets or empty data blocks that are created within the datastore.
- **Virtual network management** of Virtual networks that can be organized in network catalogs, and provide means to interconnect virtual machines. This kind of resources can be defined as IPv4, IPv6, or mixed networks, and can be used to achieve full isolation between virtual networks. Networks can be easily interconnected by using virtual routers and KVM users can also dynamically configure security groups
- **Virtual machine template management** with template catalog system that allows to register virtual machine definitions in the system, to be instantiated later as virtual machine instances.
- **Virtual machine instance management** with a number of operations that can be performed to control lifecycle of the virtual machine instances, such as migration (live and cold), stop, resume, cancel, power-off, etc.

Several reference guides are provided for more information about definition files, templates and CLI.

4.3. Create Virtual Machines

One of the most important aspects of teh cloud is the **preparation of the images** for our users. OpenNebula uses a method called contextualization to send information to the VM at boot time. Its most basic usage is to share networking configuration and login credentials with the VM so it can be configured. More advanced cases can be starting a custom script on VM boot or preparing configuration to use OpenNebula Gate.

1.1.5 Step 5. Install Advanced Components

This step is optional and only for advanced users. We recommend you familiarize with OpenNebula before installing these components.

OpenNebula brings the following advanced components:

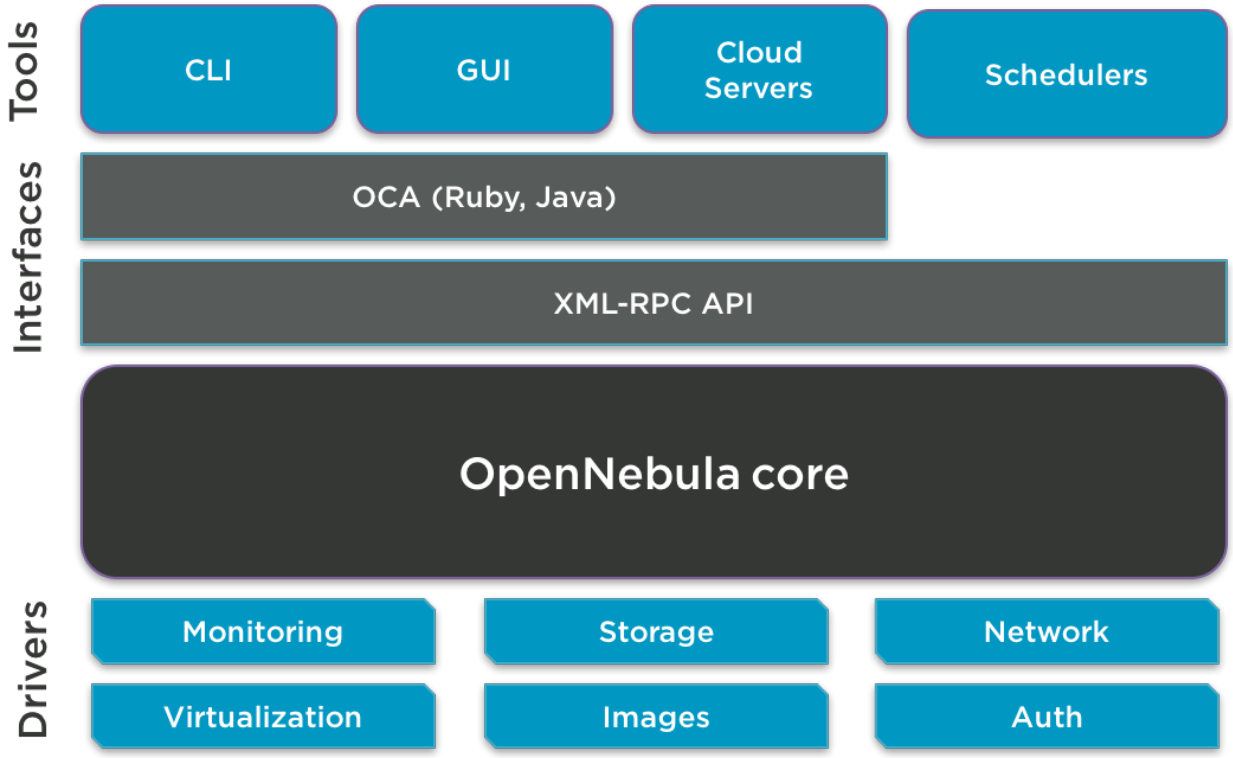
- Implementation of the EC2 Query and EBS **public cloud** interfaces.
- OneFlow allows **multi-VM application and auto-scaling** to define, execute and manage multi-tiered elastic applications, or services composed of interconnected Virtual Machines with deployment dependencies between them and auto-scaling rules.
- The datacenter federation functionality allows for the **centralized management of multiple instances of OpenNebula for scalability, isolation and multiple-site support**.
- **Application insight** with OneGate allows Virtual Machine guests to pull and push VM information from OpenNebula. Users and administrators can use it to gather metrics, detect problems in their applications, and trigger OneFlow elasticity rules from inside the VM.
- Marketplaces for sharing, provisioning and consuming cloud images. They can be seen as external datastores, where images can be easily imported, exported and shared by a federation of OpenNebula instances.
- **Cloud bursting** gives support to build a hybrid cloud, an extension of a private cloud to combine local resources with resources from remote cloud providers. A whole public cloud provider can be encapsulated as a local resource to be able to use extra computational capacity to satisfy peak demands. Out of the box connectors are shipped to support Amazon EC2 and Microsoft Azure cloudbursting.

1.1.6 Step 6. Integrate with other Components

This step is optional and only for integrators and builders.

Because no two clouds are the same, OpenNebula provides many different interfaces that can be used to interact with the functionality offered to manage physical and virtual resources.

- **Modular and extensible architecture** with customizable plug-ins for integration with any third-party data center infrastructure platform for storage, monitoring, networking, authentication, virtualziation, cloud bursting and market.
- **API for integration** with higher level tools such as billing, self-service portals... that offers all the rich functionality of the OpenNebula core, with bindings for ruby and java and XMLRPC API,
- **OneFlow API** to create, control and monitor multi-tier applications or services composed of interconnected Virtual Machines.
- **Sunstone custom routes and tabs** to extend the sunstone server.
- **Hook Manager** to trigger administration scripts upon VM state change.



1.2 OpenNebula Key Features

OpenNebula offers a **simple but feature-rich and flexible solution** to build and manage data center virtualization and enterprise clouds. This guide summarizes its key features(*). You can also refer to the *Platform Notes* included in the documentation of each version to know about the infrastructure platforms and services supported by OpenNebula.

INTERFACES FOR CLOUD CONSUMERS

- De-facto standard cloud APIs with compatibility with cloud ecosystem tools
- Simple, clean, intuitive GUI for cloud consumers to allow non-IT end users to easily create, deploy and manage compute, storage and network resources

VIRTUAL MACHINE MANAGEMENT

- Virtual infrastructure management adjusted to enterprise data centers with full control, monitoring and accounting of virtual resources

- Virtual machine image management through catalogs of disk images (termed datastores) with OS installations, persistent data sets or empty data blocks that are created within the datastore
- Virtual machine template management through catalogs of templates that allow to register virtual machine definitions in the system to be instantiated later as virtual machine instances
- Virtual machine instance management with full control of virtual machine lifecycle
- Programmable VM operations allowing users to schedule actions
- Volume hotplugging and disk snapshot capabilities and disk resizing for KVM virtual machines

VIRTUAL NETWORK MANAGEMENT

- 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...
- IPv6 support with definition site and global unicast addresses
- Virtual routers
- Security Groups to define firewall rules and apply them to KVM Virtual Machines

APPLICATION CONFIGURATION AND INSIGHT

- Automatic installation and configuration of application environments
- VM attributes can be provided by the user when the template is instantiated
- Wide range of guest operating system including Microsoft Windows and Linux
- Gain insight cloud applications so their status and metrics can be easily queried through OpenNebula interfaces and used in auto-scaling rules

MULTI-VM APPLICATION MANAGEMENT

- Automatic execution of multi-tiered (multi-VM) applications and their provision from a catalog and self-service portal
- Automatic scaling of multi-tiered applications according to performance metrics and time schedule

INTERFACES FOR ADMINISTRATORS AND ADVANCED USERS

- Powerful Command Line Interface that resembles typical UNIX commands applications
- 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

APPLIANCE MARKETPLACE

- Access to the public OpenNebula Systems Marketplace with a catalog of OpenNebula-ready cloud images
- Create your private centralized catalog (external satastore) of cloud applications (images and templates)
- Move VM images and templates across different types of datastores within the same OpenNebula instance
- Share VM images in Federation environments across several OpenNebula instances

ACCOUNTING AND SHOWBACK

- Configurable accounting system to report resource usage data and guarantee fair share of resources among users
- Easy integration with chargeback and billing platforms
- Showback capabilities to define cost associated to CPU/hours and MEMORY/hours per VM Template

MULTI-TENANCY AND SECURITY

- Fine-grained ACLs for resource allocation
- Powerful user and role management
- Administrators can groups users into organizations that can represent different projects, division...
- Integration with external identity management services
- Special authentication mechanisms for SunStone (OpenNebula GUI) and the Cloud Services (EC2)
- Login token functionality to password based logins
- Fine-grained auditing
- Support for isolation at different levels

ON-DEMAND PROVISION OF VIRTUAL DATA CENTERS

- A VDC (Virtual Data Center) is a fully-isolated virtual infrastructure environment where a Group of users, optionally under the control of the group admin, can create and manage compute and storage capacity
- There is a pre-configured Sunstone view for group admins

CAPACITY AND PERFORMANCE MANAGEMENT

- Host management with complete functionality for the management of the virtualziation nodes in the cloud
- Dynamic creation of Clusters as pools of hosts that share datastores and virtual networks for load balancing, high availability, and high performance computing
- customizable and highly scalable monitoring system and also can be integrated with external data center monitoring tools.
- Powerful and flexible scheduler for the definition of workload and resource-aware allocation policies such as packing, striping, load-aware, affinity-aware...
- Resource quota management to track and limit computing, storage and networking resource utilization
- 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 KVM VM types or users
- PCI passthrough available for KVM VMs that need consumption of raw GPU devices

FEDERATED CLOUD ENVIRONMENTS

- 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

HIGH AVAILABILITY AND BUSINESS CONTINUITY

- High availability architecture in active-passive configuration
- Persistent database backend with support for high availability configurations
- Configurable behavior in the event of host or KVM VM failure to provide easy to use and cost-effective failover solutions

CLOUD BURSTING

- Build a hybrid cloud to combine your local resources with resources from remote cloud provider and use extra computational capacity to satisfy peak demands

PLATFORM

- Fully platform independent

- Hypervisor agnostic with broad hypervisor support (KVM and VMware vCenter) and centralized management of environments with multiple hypervisors
- *Broad support for commodity and enterprise-grade hypervisor, monitoring, storage, networking and user management services*
- Packages for major Linux distributions

CUSTOMIZATION 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: storage, monitoring, networking, authentication, virtualziation, cloud bursting and market
- API for integration with higher level tools such as billing, self-service portals...
- Hook manager to trigger administration scripts upon VM state change
- Sunstone custom routes and tabs to extend the sunstone server
- OneFlow API to create, control and monitor multi-tier applications or services composed of interconnected Virtual Machines.
- OpenNebula Add-on Catalog with components enhancing the functionality provided by OpenNebula
- Configuration and tuning parameters to adjust behavior of the cloud management instance to the requirements of the environment and use cases

LICENSING

- Fully open-source software released under Apache license

INSTALLATION AND UPGRADE PROCESS

- *Configurable to deploy public, private and hybrid clouds*
- All key functionalities for enterprise cloud computing, storage and networking in a single install
- Long term stability and performance through a *single integrated patching and upgrade process*
- Automatic import of existing VMs running in local hypervisors and public clouds for hybrid cloud computing
- Optional building from source code
- System features a small footprint, less than 10Mb

QUALITY ASSURANCE

- Internal quality assurance process for functionality, scalability, performance, robustness and stability
- Technology matured through an active and engaged large community
- Scalability, reliability and performance tested on many massive scalable production deployments consisting of hundreds of thousands of cores and VMs

PRODUCT SUPPORT

- Best-effort community support
- SLA-based commercial support directly from the developers
- Integrated tab in Sunstone to access OpenNebula Systems professional support

(*) *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 Glossary

1.3.1 OpenNebula Components

- **Front-end**: Machine running the OpenNebula services.
- **Host**: Physical machine running a supported hypervisor. See the Host subsystem.
- **Cluster**: Pool of hosts that share datastores and virtual networks. Clusters are used for load balancing, high availability, and high performance computing.
- **Datastore**: Storage medium used as disk images repository or to hold images for running VMs.
- **Sunstone**: OpenNebula web interface. Learn more about Sunstone
- **Self-Service** OpenNebula web interfaced towards the end user. It is implemented by configuring a user view of the Sunstone Portal.
- **EC2 Service**: Server that enables the management of OpenNebula with EC2 interface. Learn more about EC2 Service.
- **OCA**: OpenNebula Cloud API. It is a set of libraries that ease the communication with the XML-RPC management interface. Learn more about ruby and java APIs.

1.3.2 OpenNebula Resources

- **Template**: Virtual Machine definition. These definitions are managed with the onetemplate command.
- **Image**: Virtual Machine disk image, created and managed with the oneimage command.
- **Virtual Machine**: Instantiated Template. A Virtual Machine represents one life-cycle, and several Virtual Machines can be created from a single Template. Check out the VM management guide.
- **Virtual Network**: A group of IP leases that VMs can use to automatically obtain IP addresses. See the Networking subsystem.
- **Virtual Data Center (VDC)**: Defines an assignment of one or several Groups to a pool of Physical Resources. Typically this pool of Physical Resources consists of resources from one or several Clusters that could belong to different Zones or public external clouds for hybrid cloud computing.
- **Zone**: A group of interconnected physical hosts with hypervisors controlled by the same OpenNebula.

1.3.3 OpenNebula Management

- **ACL**: Access Control List. Check the managing ACL rules guide.
- **oneadmin**: Special administrative account. See the Users and Groups guide.
- **User**: An OpenNebula user account.
- **Group**: A group of Users.

- **Federation**: Several OpenNebula instances can be configured as zones.

CHAPTER

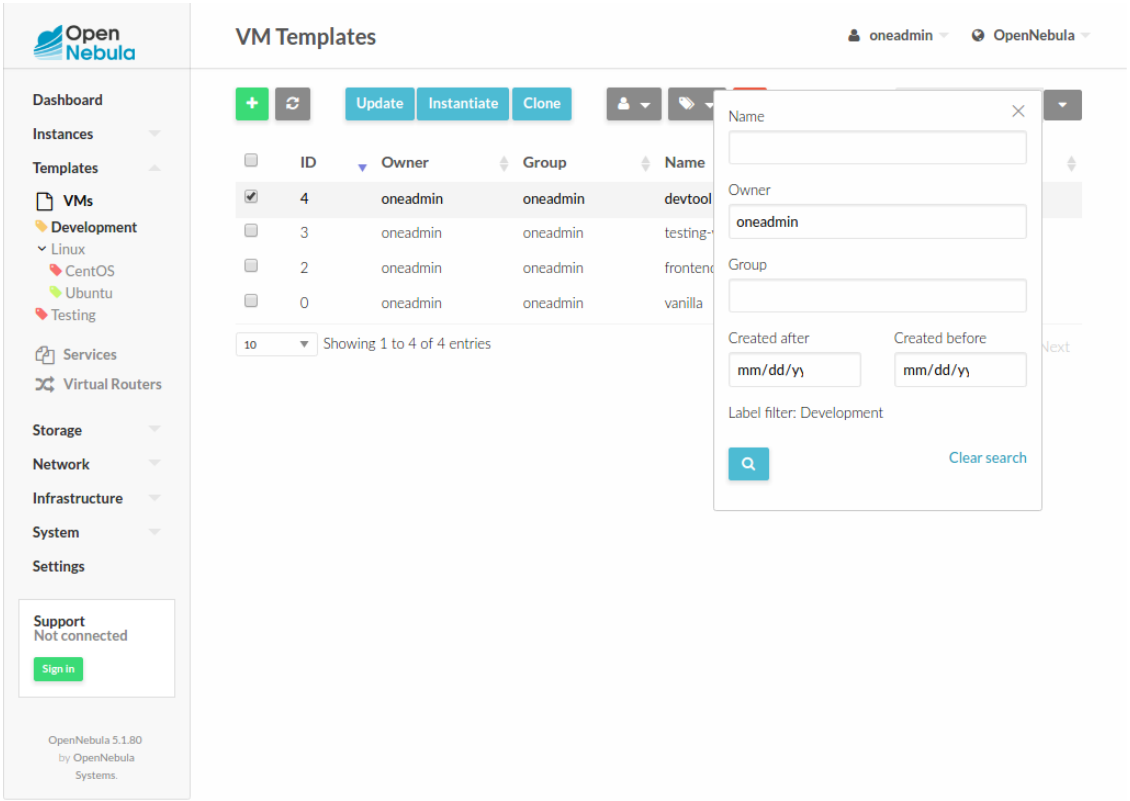
TWO

RELEASE NOTES 5.2.1

2.1 What's New in 5.2

OpenNebula 5.2 (Excession) is the second release of the OpenNebula 5 series. A significant effort has been applied in this release to stabilize features introduced in 5.0 Wizard, while keeping an eye in implementing those features more demanded by the community.

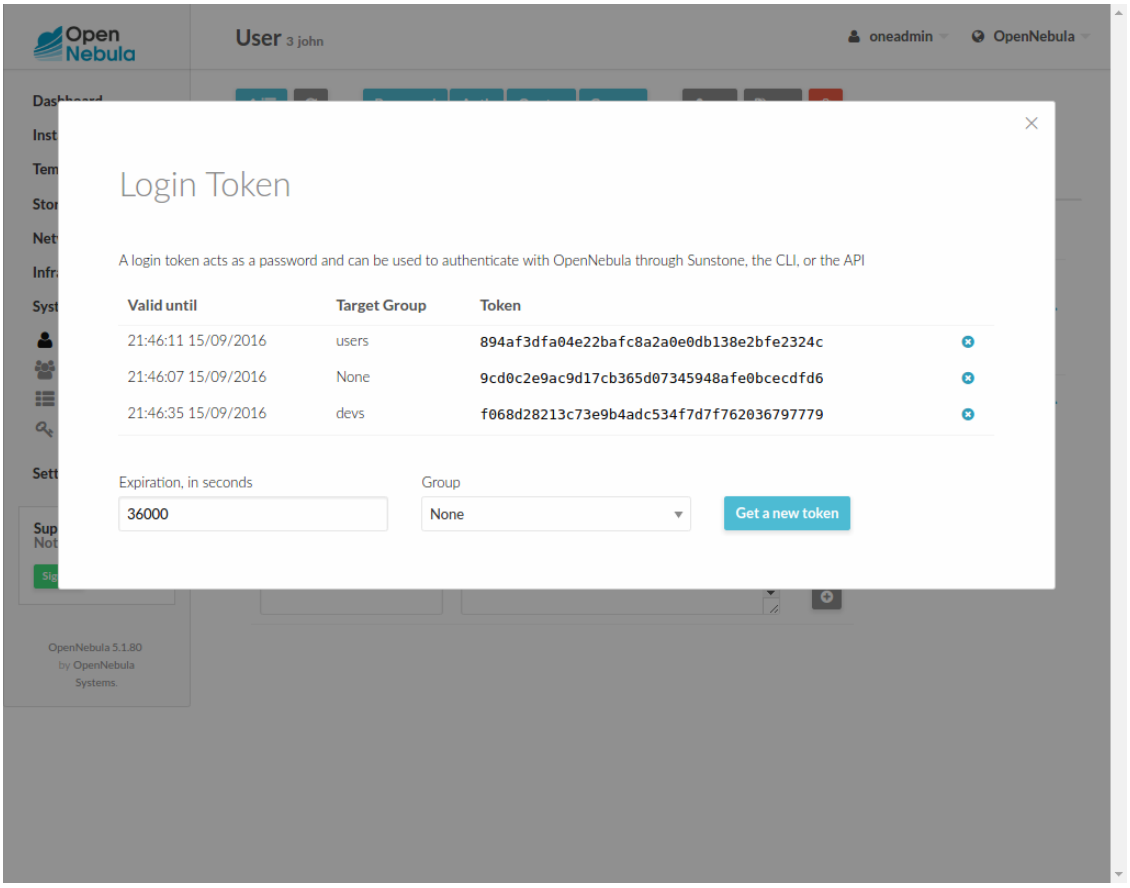
As usual almost every component of OpenNebula has been reviewed to target usability and functional improvements, trying to keep API changes to a minimum to avoid disrupting ecosystem components. Also, new components have been added to enhance the OpenNebula experience.



One important new module is the IPAM subsystem. In order to foster SDN integration, a important step is being able to integrate OpenNebula with existing IPAM modules, in those cases where outsourcing of IP management is required in the datacenter. Fitting in the OpenNebula architecture design principles, the IPAM subsystem interacts with IPAM

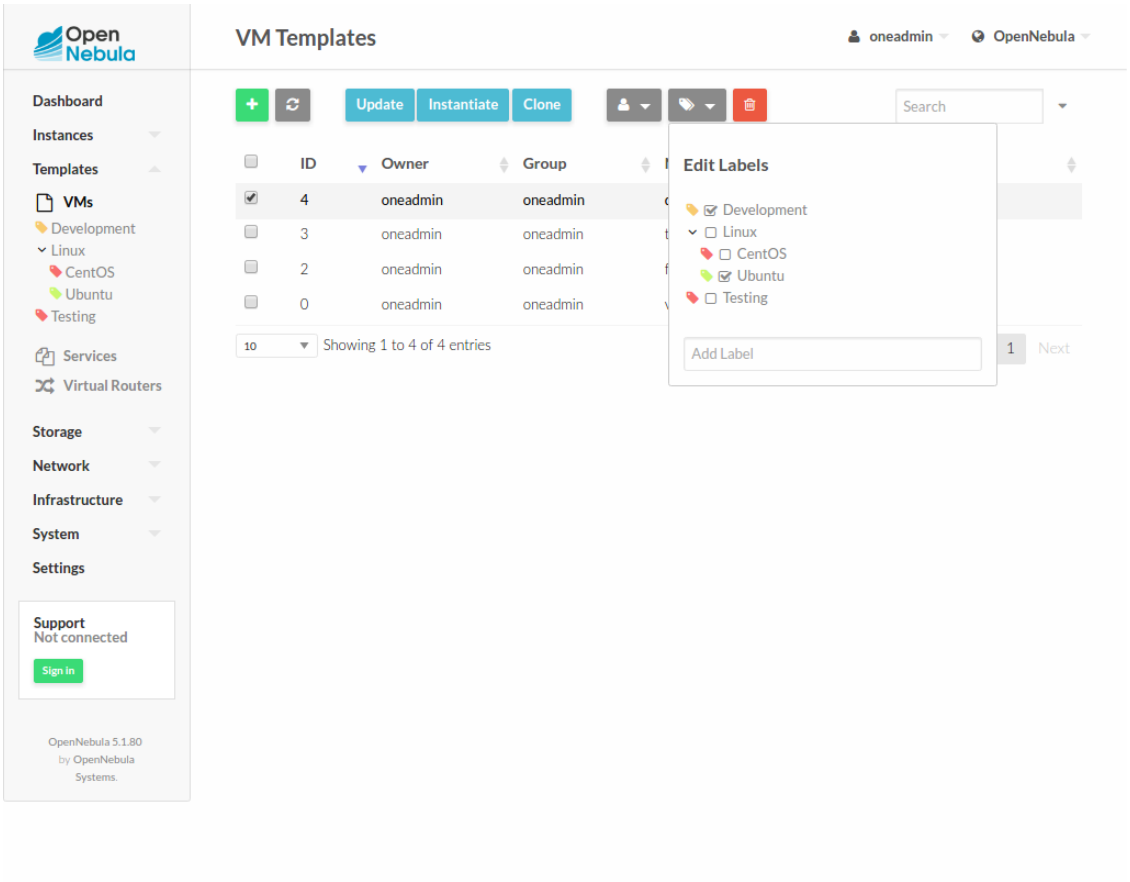
servers using drivers, and as such a IPAM driver lets you delegate IP lease management to an external component. This way you can coordinate IP use with other virtual or bare metal servers in your datacenter. No default integration is provided, but rather to effectively use an external IPAM you need to develop four action scripts that hook on different points of the IP network/lease life-cycle.

Another great addition in Excession is the ability to use group bound tokens. The goal is to be able to use OpenNebula for different projects, which are identified with different groups. For instance, the same user can use OpenNebula for "WebDevelopment" project and a "BioResearch" one, for instance. This user can request a couple of tokens tied to each of these groups. Upon login with the "WebDevelopment" token, she will only be seeing resources from that particular project, and all new resources (VMs, images, networks) will be created within that group, isolating them from the "BioResearch" group. This feature is available both in the CLI and Sunstone, with helpers and dialogs to create, maintain and use the tokens.



All the OpenNebula drivers have been improved for robustness. For instance, a new default timeout (which is configurable) has been defined to identify hanging operations and kill crashed processes. In this regard, the EC2 drivers has also been thoroughly revisited, being updated to the v2 of the aws ruby gem, ensuring compatibility with all Amazon EC2 regions. Error handling has been improved as well in the EC2 driver, adding operation retries to circumvent those situations where the EC2 API is not consistent, and adding improved logging.

Sunstone is the face of OpenNebula for both administrators and users, and hence a constant target of enhancements to improve usability. Excession brings to the cloud table stabilized features that were introduced in the Wizard maintenance releases, like for instances advanced searches (that now are maintained regardless of tab switching), labels colors and ergonomics, improved vCenter dialogs and import tables (now with feature à la Gmail), hyperlinks to access resources displayed in the info tabs, and many other minor improvements.



There are many other improvements in 5.2 like revamped group mapping in LDAP authentication -now being dynamic mapping-, limiting bandwidth per VM network interface in both KVM and vCenter, rollback mechanism in failed migrate operations, significantly improved fault tolerant hook -to provide high availability at the VM level-, improved driver timeout, vCenter storage functionality wrinkles ironed out, more robust Ceph drivers -for instance, in volatile disks-, improved SPICE support, improvements in ebtables and Open vSwitch drivers, multiple CLI improvements -imrpved onedb patch, password handling in onevcenter command, default columns reviewed in all commands- and much more. As with previous releases, it is paramount to the project to help build and maintain robust private, hybrid and public clouds with OpenNebula, fixing reported bugs and improving general usability.

This OpenNebula release is named after the Ian M. Banks novel, a recommended read, as well as having a fitting slang meaning, "something so technologically superior that it appears as magic to the viewer.". We are confident that OpenNebula, if not really appearing as magic, at least solves elegantly your IaaS needs.

OpenNebula 5.2 Excession 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 5.2 (a detailed list of changes can be found here):

2.1.1 OpenNebula Core

- **Improved FT hook**, to enhancing logging and fencing mechanisms integration in the host on error hook.
- **Project and group management**, adding authorization tokens to include session information, to allow different group/project sessions with the same user, paired with a new parameter in the one.user.allocate API call and new filter flag.
- **Better template management in marketplaces**, with rethinked restricted attributes.

- **Rollback capabilities**, in the migrate operation.
- **Update group information**, if driver (such as LDAP) provides it.
- **Improved range definitions** for the vlan driver
- **Allow migration** between clusters provided they share datastores between them.
- **Outsource IP management** with the new :ref:IPAM subsystem <*ipam*>.
- **Allow to override** of EMULATOR attribute placing it in the VM template.

2.1.2 OpenNebula Drivers :: Storage

- **Files are copied directly from source to target hypervisor** when migrating over SSH.

2.1.3 OpenNebula Drivers :: Virtualization

- **Add timeouts** for driver actions.
- **Stop execution** of drivers if a pipe fails
- **Improved driver** for EC2 integration, now using aws-sdk v2 ruby gem and with double checks and retries in EC2 API methods and responses, as well as better error logging

2.1.4 OpenNebula Drivers :: Networking

- **Limit network consumption**, per VM network interface.

2.1.5 OpenNebula Drivers :: Marketplace

- **Enable access behind HTTP proxy** for marketplaces.

2.1.6 Scheduler

- **Improved datastore capacity tests**, for datastore space in scheduler.
- **Add a new sched_message** when a VM cannot be dispatched.

2.1.7 Sunstone

- **Better display**, for labels as well as automatic color assignment.
- **Name display** when instantiating a Virtual Router template.
- **required inputs and better datatables** in vCenter import dialogs.
- **Compatibility with browser history**, now is possible to use the back button
- **Better user workflow**, wizards now return to the individual view
- **Improved VM capacity graphs**, now displaying the allocated value for better understanding of resource consumption.
- **Hyperlinks for related resource**, like for instance clicking on an image ID inside a VM disk attribute goes directly to the image in the image tab. Nice!

2.1.8 Command Line Interface

- **Improved onedb command**, onedb patch now accepts arguments.
- **Better password prompt** when doing oneuser login.
- **vCenter import command improved**, onevcenter now asks for password in prompt if not provided in arguments.
- **Improved default columns** in commands output.
- **New filter flag**, to aid in project and group management through the CLI.

2.1.9 vCenter

- **Support for clustered Datastores**, SDRS clustered datastores are now visible in vCenter
- **Better Sunstone support**, with host dropdown option in DS creation and image import
- **Network monitoring**, now vCenter VMs network traffic is accounted for in OpenNebula.

2.2 Resolved Issues in 5.2.1

New functionality for 5.2.1 has been introduced:

- Improvements in SDR clustered vcenter datastores.
- Check if bridge contains incorrect vlan.
- Strengthen deploy checks to avoid issues with EC2 API lag.

The following issues has been solved in 5.2.1:

- Error when terminating a VM with more than ten interfaces.
- onedb fsck breaks with volatile disks.
- TAGS attribute are not being set in EC2.
- Cannot import two vCenter networks with the same name in two clusters.
- Ceph persistent image “loses” changes.
- Fix NETTX and NETRX negative values, polling time and accumulate values.
- Fix situations that would cause vCenter NICs not being removed.
- Add the possibility to introduce a empty ssh key in Sunstone.
- Sunstone Settings field can be disabled.

2.3 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.

2.3.1 Certified Components Version

Front-End Components

Component	Version	More information
RedHat Enterprise Linux	7.0	Front-End Installation
Ubuntu Server	14.04 (LTS), 16.04 (LTS)	Front-End Installation
CentOS	7.0	Front-End Installation
Debian	8	Front-End Installation
MariaDB or MySQL	Version included in the Linux distribution	MySQL Setup
SQLite	Version included in the Linux distribution	Default DB, no configuration needed
Ruby Gems	Versions installed by packages and install_gems utility	front-end installation
Corosync+Pacemaker	Version included in the Linux distribution	Front-end HA Setup

vCenter Nodes

Component	Version	More information
vCenter	5.5/6.0, managing ESX 5.5/6.0	vCenter Node Installation

KVM Nodes

Component	Version	More information
RedHat Enterprise Linux	7.0	KVM Driver
Ubuntu Server	14.04 (LTS) , 16.04	KVM Driver
CentOS/RHEL	7.0	KVM Driver
Debian	8	KVM Driver
KVM/Libvirt	Support for version included in the Linux distribution. For CentOS/RedHat the packages from `qemu-ev` are used.	KVM Node Installation

Open Cloud Networking Infrastructure

Component	Version	More information
ebtables	Version included in the Linux distribution	Ebtables
8021q kernel module	Version included in the Linux distribution	802.1Q VLAN
Open vSwitch	Version included in the Linux distribution	Open vSwitch
iproute2	Version included in the Linux distribution	VXLAN

Open Cloud Storage Infrastructure

Component	Version	More information
iSCSI	Version included in the Linux distribution	LVM Drivers
LVM2	Version included in the Linux distribution	LVM Drivers
Ceph	Hammer (LTS) v0.94	The Ceph Datastore

Authentication

Component	Version	More information
net-ldap ruby library	0.12.1	LDAP Authentication
openssl	Version included in the Linux distribution	x509 Authentication

Cloud Bursting

Component	Version	More information
aws-sdk	1.66	Amazon EC2 Driver
azure	0.6.4	Azure Driver

Note: Generally for all Linux platforms, it is worth noting that gems should be installed with the install_gems, avoiding the platform's package version.

2.3.2 Frontend Platform Notes

The following applies to all Front-Ends:

- XML-RPC 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`.
- Only **ruby versions >= 1.9.3 are supported**.

Ubuntu 14.04 Platform Notes

Package ruby-ox shouldn't be installed as it contains a version of the gem incompatible with the CLI

CentOS 7.0 Platform Notes

This distribution lacks some packaged ruby libraries. This makes some components unusable until they are installed. In the front-end, just after package installation these commands 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 remove the line that removes `/var/tmp`.

There is a bug in libvirt that the prevents the use of the save/restore mechanism if `cpu_model` is set to `'host-passthrough'` via `RAW`. The work around if needed is described in this issue.

Debian 8

Make sure that the packages `ruby-treetop` and `treetop` are not installed before running `install_gems`, as the version of `treetop` that comes packaged in Debian 8 is incompatible with OpenNebula. **OneFlow** requires a version >= 1.6.3 for treetop, packages distributed with Debian 8 includes an older version (1.4.5).

2.3.3 Nodes Platform Notes

The following items apply to all distributions:

- Since OpenNebula 4.14 there is a new monitoring probe that gets information about PCI devices. By default it retrieves all the PCI devices in a host. To limit the PCI devices that it gets info and appear in `onehost show` refer to kvm_pci_passthrough.
- When using qcow2 storage drivers you can make sure that the data is written to disk when doing snapshots setting its `cache` parameter to `writethrough`. This change will make writes slower than other cache modes but safer. To do this edit the file `/etc/one/vmm_exec/vmm_exec_kvm.conf` and change the line for `DISK`:

```
DISK = [ driver = "qcow2", cache = "writethrough" ]
```

CentOS/RedHat 7.0 Platform Notes

Ruby Dependencies

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 7 repositories to install ruby dependencies.

Libvirt Version

The libvirt/qemu packages used in the testing infrastructure are the ones in the `qemu-ev` repository. To add this repository you can install the following packages:

```
# yum install centos-release-qemu-ev
# yum install qemu-kvm-ev
```

Disable PolicyKit for Libvirt

It is recommended that you disable PolicyKit for Libvirt:

```
$ cat /etc/libvirt/libvirtd.conf
...
auth_unix_ro = "none"
auth_unix_rw = "none"
unix_sock_group = "oneadmin"
unix_sock_ro_perms = "0770"
unix_sock_rw_perms = "0770"
...
```

2.3.4 Unsupported Platforms Notes

Warning: Use the following distributions at your own risk. They are not officially supported by OpenNebula.

CentOS 6.5 Usage Platform Notes

- As a front-end, 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.
- As a node, to accomplish disk hotplugging:
  - to accomplish disk hotplugging, disks need to be attached through SCSI, so their images should have a DEV_PREFIX="sd"
  - to accomplish disk hotplugging, 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 libvirt/qemu attach/detach nic␣
↪functionality that prevents the reuse of net IDs. This means that after a␣
↪successful attach/detach NIC, a new attach will fail.
```

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
```

Installing on Devuan

Packages for Devuan Jessie 1.0 Beta are provided by Alberto Zuin. You can download them at:

http://downloads.opennebula.org/extra/packages/devuan/

2.4 Compatibility Guide

This guide is aimed at OpenNebula 5.0.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 *section*

Visit the *Features list* and the Release Notes for a comprehensive list of what's new in OpenNebula 5.2.

2.4.1 OpenNebula Administrators

OpenNebula Daemon

The attributes `VNC_PORTS/RESERVED` and `VLAN_IDS/RESERVED` attribute in oned.conf now accept port ranges:

```
#  VNC_PORTS: VNC port pool for automatic VNC port assignment, if possible the
#  port will be set to ``START`` + ``VMID``
#   start   : first port to assign
#   reserved: comma separated list of ports or ranges. Two numbers separated by
#   a colon indicate a range.

VNC_PORTS = [
    START    = 5900
    RESERVED = "6800, 6801, 6810:6820, 9869"
]

#  VLAN_IDS: VLAN ID pool for the automatic VLAN_ID assignment. This pool
#  is for 802.1Q networks (Open vSwitch and 802.1Q drivers). The driver
#  will try first to allocate VLAN_IDS[START] + VNET_ID
#     start: First VLAN_ID to use
#     reserved: Comma separated list of VLAN_IDs or ranges. Two numbers
#     separated by a colon indicate a range.

VLAN_IDS = [
    START    = "2",
    RESERVED = "0, 1, 4095"
]
```

Fault Tolerance Hook

The fault tolerance hook now has a new option, `-u`:

```
#*******************************************************************************
# Fault Tolerance Hooks
#*******************************************************************************
# This hook is used to perform recovery actions when a host fails.
# Script to implement host failure tolerance
#   One of the following modes must be chosen
#           -m resched VMs to another host. (Only for images in shared storage!)
#           -r recreate VMs running in the host. State will be lost.
#           -d delete VMs running in the host
#
#   Additional flags
#           -f resubmit suspended and powered off VMs (only for recreate)
#           -p <n> avoid resubmission if host comes back after n monitoring
#                 cycles. 0 to disable it. Default is 2.
#           -u disables fencing. Fencing is enabled by default. Don't disable it
#                 unless you are very sure about what you're doing
#*******************************************************************************
```

Virtual Machine Management

New recovery option for a Virtual Machine is these states:

- `PROLOG_MIGRATE_FAILURE`
- `PROLOG_MIGRATE_POWEROFF_FAILURE`
- `PROLOG_MIGRATE_SUSPEND_FAILURE`
- `PROLOG_MIGRATE_UNKNOWN_FAILURE`

If a migration to another Host fails the administrator can fix the situation and:

- **[NEW in 5.2]** Force OpenNebula to assume the VM is running in the previous Host (where the migration was started):

```
$ onevm recover <id> --failure
```

- Force OpenNebula to assume the VM is running in the destination Host:

```
$ onevm recover <id> --success
```

- Retry the migration:

```
$ onevm recover <id> --retry
```

User Tokens

The login token functionality has been improved with two main differences:

- More than one token can be defined per user
- Each token can have an "effective Group ID"

Instead of the `oneuser login` command, the tokens are now managed with these new sub-commands:

- `oneuser token-create [<username>]`
- `oneuser token-set [<username>]`
- `oneuser token-delete [<username>] <token>`
- `oneuser token-delete-all <username>`

Read more about login tokens in the Managing Users documentation.

LDAP Group Management

The LDAP drivers were capable of creating new users in a set of configured groups. Now in OpenNebula 5.2 the user's groups will be updated after the user creation if the LDAP driver reports a different list of Group IDs.

Read more in the LDAP Authentication documentation.

Migration Across Clusters

Before OpenNebula 5.2 a VM migration could only be performed if the current Host was in the same Cluster as the new Host. This ensured that the destination Host had access to the same infrastructure requirements: Datastores and Virtual Networks.

Now the requirements have changed to allow the migration if the destination Host is in a Cluster that contains all the Datastores and Virtual Networks required by the VM. In an homogeneous infrastructure this allows greater flexibility, but there is a special case where this change could be problematic:

If the Clusters separate incompatible Hosts (incompatible hypervisor versions, or hardware architecture) but contain the same set of Datastores and Virtual Networks, the migration could fail. This is specially important when the `onevm resched` command is used, as the scheduler now will decide that all those incompatible Hosts are eligible for the migration.

2.4.2 Developers and Integrators

IM and VM Drivers

Each `IM_MAD` and `VM_MAD` defined in oned.conf can now define a timeout with the `-w` parameter:

```
-w  Timeout in seconds to execute external commands (default unlimited)
```

IPAM Drivers

There is a new kind of driver to interact with existing IPAM modules. Read more about it in the IPAM driver documentation.

Authentication Drivers

The authentication drivers defined in oned.conf now have 2 extra attributes that define their behavior, `DRIVER_MANAGED_GROUPS` and `MAX_TOKEN_TIME`.

```
#*******************************************************************************
# Authentication Driver Behavior Definition
#*******************************************************************************
# The configuration for each driver is defined in AUTH_MAD_CONF. These
# values must not be modified since they define the driver behavior.
#   name            : name of the auth driver
#   password_change : allow the end users to change their own password. Oneadmin
#                     can still change other user's passwords
#   driver_managed_groups : allow the driver to set the user's group even after
#                     user creation. In this case addgroup, delgroup and chgrp
#                     will be disabled, with the exception of chgrp to one of
#                     the groups in the list of secondary groups
#   max_token_time  : limit the maximum token validity, in seconds. Use -1 for
#                     unlimited maximum, 0 to disable login tokens
#*******************************************************************************

AUTH_MAD_CONF = [
    NAME = "core",
    PASSWORD_CHANGE = "YES",
    DRIVER_MANAGED_GROUPS = "NO",
    MAX_TOKEN_TIME = "-1"
]

AUTH_MAD_CONF = [
    NAME = "ldap",
    PASSWORD_CHANGE = "YES",
    DRIVER_MANAGED_GROUPS = "YES",
```

```
    MAX_TOKEN_TIME = "86400"
]
```

XML-RPC API

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

- Changed api calls:
  - `one.user.login`: New parameter EGID, effective GID to use with this token. To use the current GID and user groups set it to -1
  - `one.user.allocate`: New parameter gids, array of Group IDs. To create a new User setting the main and secondary groups directly
  - `one.*pool.info`: New filter flag **-4**, to request resources that belong to the user's primary group only.

2.5 Known Issues

2.5.1 CLI

- #3037 Different ruby versions need different time formats
- #3337 Wrong headers when output is piped for oneacct and oneshowback

2.5.2 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
- #3888 An image can be used twice by a VM, but this breaks the used/ready logic
- #4563 onetemplate instantiate –persistent uses always the same name
- #4154 after cold migration custom vnc password is lost
- #4015 Showback records authorization relies on current VM ownership, instead of Showback record owner
- #3020 OpenNebula should check the available space in the frontend before doing an undeploy
- #2998 Deleting an image deletes the record from the database even if the delete fails
- #1494 Newlines wrongly interpreted in multiline context variables with template variables

2.5.3 Drivers - Network

- #3093 Review the Open vSwitch flows
- #2961 review nic attach with 802.1Q
- #4005 NIC defaults not honoured in attach NIC
- #4821 No traffic shaping in attached NICs

2.5.4 Drivers - Storage

- #1573 If an image download fails, the file is never deleted from the datastore
- #3929 CEPH_HOST not IPv6 friendly
- #3727 Downloads can fail, but still not return in error
- #3705 Images downloaded from the Marketplace to Ceph are left with DRIVER=qcow2

2.5.5 Drivers - VM

- #4648 Delete operation leaves a poweroff instance registered in vCenter
- #3060 Trim/discard. In order to use this functionality, KVM requires the virtio-scsi controller. This controller is not (yet) supported. In order to add it, you need to add this RAW snippet to your template: `<controller type='scsi' index='0' model='virtio-scsi'></controller>`
- #4540 Import vCenter images size may be 0
- #4335 vCenter password cannot be longer than 22 characters
- #4514 Spaces not allowed in SOURCE image attribute

2.5.6 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
- #2155 Scheduled policy start_time cannot be defined as a POSIX time number

2.5.7 Scheduler

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

2.5.8 Sunstone

- #1877 if syslog enabled disable the logs tab in the VM detailed view
- #3796 sunstone ignores the no_proxy environment variable
- #4567 VM Template create wizard: vCenter option should not allow to create volatile disks
- #3902 LIMIT_MB is not used to calculate the available DS storage
- #3692 Sunstone image upload - not enough space
- #2867 Sunstone template update does not select images without User Name
- #2801 Template update: placement does not select the hosts/clusters
- #4574 vCenter VMs VNC after stop/resume does not work
- #4652 noVNC mouse doesn't work when a touchscreen is present

2.5.9 Context

- #4568 Context is not regenerated for vCenter

2.5.10 vCenter

- #4693 vlan_id is only imported for networks of type DistributedVirtualPortgroup
- #4699 Vcenter does not honor vmm_exec_vcenter.conf

2.6 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.

The group token functionality and dynamic LDAP group mapping into OpenNebula groups were funded by BlackBerry in the context of the Fund a Feature Program.

CHAPTER

THREE

UPGRADING

3.1 Overview

Keeping your OpenNebula up-to-date is very important, as you will receive the latest functionality and more importantly, the latest security patches. It is possible to upgrade to the latest OpenNebula release from earlier versions.

3.1.1 Hypervisor Compatibility

The upgrade procedure can be followed regardless of the hypervisor.

3.1.2 How Should I Read This Chapter

You only need to read this chapter if you are upgrading OpenNebula to a newer release. Make sure you have read the *Release Notes* and particularly the *Compatibility* section first.

Upgrading is a sequential procedure. The system will upgrade from the currently installed release to the latest release going through each release (if any). Therefore it's important to read each section.

After the upgrade procedure you can continue using your upgraded OpenNebula Cloud.

3.2 Upgrading from OpenNebula 5.2.x

This section describes the installation procedure for systems that are already running a 5.2.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 5.2.

3.2.1 Upgrading a Federation

If you have two or more 5.2.x OpenNebulas working as a Federation, you can upgrade each one independently. Zones with an OpenNebula from the 5.2.x series can be part of the same federation, since the shared portion of the database is compatible.

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.

3.2.2 Upgrading from a High Availability deployment

The recommended procedure to upgrade two OpenNebulas configured in HA is to follow the upgrade procedure in a specific order. Some steps need to be executed in both servers, and others in just the active node. For the purpose of this section, we will still refer to the *active node* as such even after stopping the cluster, so we run the single node steps always in the same node:

- *Preparation* in the active node.
- *Backup* in the active node.
- Stop the cluster in the active node: `pcs cluster stop`.
- *Installation* in both nodes. Before running `install_gems`, run `gem list > previous_gems.txt` so we can go back to those specific `sinatra` and `rack` gems if the `pcsd` refuses to start.
- *Configuration Files Upgrade* in the active node.
- *Database Upgrade* in the active node.
- *Check DB Consistency* in the active node.
- *Reload Start Scripts in CentOS 7* in both nodes.
- Start the cluster in the active node.

3.2.3 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.

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.

3.2.4 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')
```

3.2.5 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.

Note: If executing `install_gems` you get a message asking to overwrite files for aws executables you can safely answer "yes".

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 5.2 to your setup. If for any reason you plan to preserve your current `oned.conf` file, read the complete oned.conf reference for 5.2.

3.2.6 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.

3.2.7 Database Upgrade

The upgrade from any previous 5.2.x version does not require a database upgrade, the database schema is compatible.

3.2.8 Check DB Consistency

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

First, move the 5.2.x 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
```

3.2.9 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
```

3.2.10 Update the Drivers

You should be able now to start OpenNebula as usual, running `service opennebula start` as `root`. At this point, as `oneadmin` user, 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.

3.2.11 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"
```

3.2.12 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.

3.2.13 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.2.14 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');
```

3.3 Upgrading from OpenNebula 5.0.x

This section describes the installation procedure for systems that are already running a 5.0.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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 5.2.

3.3.1 Upgrading a Federation

If you have two or more 5.0.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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.3.2 Upgrading from a High Availability deployment

The recommended procedure to upgrade two OpenNebulas configured in HA is to follow the upgrade procedure in a specific order. Some steps need to be executed in both servers, and others in just the active node. For the purpose of this section, we will still refer to the *active node* as such even after stopping the cluster, so we run the single node steps always in the same node:

- *Preparation* in the active node.
- *Backup* in the active node.
- Stop the cluster in the active node: `pcs cluster stop`.
- *Installation* in both nodes. Before running `install_gems`, run `gem list > previous_gems.txt` so we can go back to those specific `sinatra` and `rack` gems if the `pcsd` refuses to start.
- *Configuration Files Upgrade* in the active node.
- *Database Upgrade* in the active node.
- *Check DB Consistency* in the active node.
- *Reload Start Scripts in CentOS 7* in both nodes.
- Start the cluster in the active node.

3.3.3 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.

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.

3.3.4 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')
```

3.3.5 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.

Note: If executing `install_gems` you get a message asking to overwrite files for aws executables you can safely answer "yes".

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 5.2 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 5.0 and 5.2 versions.

3.3.6 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.

3.3.7 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 4.11.80 : 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
  ...

>>> Running migrators for local tables
  ...
  > 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
```

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.

3.3.8 Check DB Consistency

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

First, move the 5.0.x 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
```

3.3.9 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.

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

- 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
```

3.3.10 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
```

3.3.11 Update the Drivers

You should be able now to start OpenNebula as usual, running `service opennebula start` as `root`. At this point, as `oneadmin` user, 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.

3.3.12 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.

3.3.13 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.3.14 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');
```

3.4 Upgrading from OpenNebula 4.14.x

This section describes the installation procedure for systems that are already running a 4.14.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

3.4.1 Upgrading a Federation

If you have two or more 4.14.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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.4.2 Upgrading from a High Availability deployment

The recommended procedure to upgrade two OpenNebulas configured in HA is to follow the upgrade procedure in a specific order. Some steps need to be executed in both servers, and others in just the active node. For the purpose of this section, we will still refer to the *active node* as such even after stopping the cluster, so we run the single node steps always in the same node:

- *Preparation* in the active node.
- *Backup* in the active node.
- Stop the cluster in the active node: `pcs cluster stop`.
- *Installation* in both nodes. Before running `install_gems`, run `gem list > previous_gems.txt` so we can go back to those specific `sinatra` and `rack` gems if the `pcsd` refuses to start.
- *Configuration Files Upgrade* in the active node.
- *Database Upgrade* in the active node.
- *Check DB Consistency* in the active node.
- *Reload Start Scripts in CentOS 7* in both nodes.
- Start the cluster in the active node.

3.4.3 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- 802.1Q
- dummy
- ebtables
- fw
- ovswitch
- vxlan

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.

3.4.4 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')
```

3.4.5 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.

Note: If executing `install_gems` you get a message asking to overwrite files for aws executables you can safely answer "yes".

It is highly recommended **not to keep** your current `oned.conf`, and update the `oned.conf` file shipped with OpenNebula 5.2 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 4.14 and 5.2 versions.

3.4.6 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.

3.4.7 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 4.11.80 : 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
  ...

>>> Running migrators for local tables
  ...
  > 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
```

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.

3.4.8 Check DB Consistency

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

First, move the 4.14 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
```

3.4.9 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 two new tables, `marketplace_pool` and `marketplaceapp_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
replicate-do-table  = opennebula.marketplace_pool
replicate-do-table  = opennebula.marketplaceapp_pool

# 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
```

3.4.10 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
```

3.4.11 Update the Drivers

You should be able now to start OpenNebula as usual, running `service opennebula start` as `root`. At this point, as `oneadmin` user, 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.

3.4.12 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.4.13 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.

3.4.14 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.

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

3.4.15 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');
```

3.5 Upgrading from OpenNebula 4.12.x

This section describes the installation procedure for systems that are already running a 4.12.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.14, 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

3.5.1 Upgrading a Federation

If you have two or more 4.12.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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.5.2 Upgrading from a High Availability deployment

The recommended procedure to upgrade two OpenNebulas configured in HA is to follow the upgrade procedure in a specific order. Some steps need to be executed in both servers, and others in just the active node. For the purpose of this section, we will still refer to the *active node* as such even after stopping the cluster, so we run the single node steps always in the same node:

- *Preparation* in the active node.
- *Backup* in the active node.
- Stop the cluster in the active node: `pcs cluster stop`.
- *Installation* in both nodes. Before running `install_gems`, run `gem list > previous_gems.txt` so we can go back to those specific `sinatra` and `rack` gems if the `pcsd` refuses to start.
- *Configuration Files Upgrade* in the active node.
- *Database Upgrade* in the active node.
- *Check DB Consistency* in the active node.
- *Reload Start Scripts in CentOS 7* in both nodes.
- Start the cluster in the active node.

3.5.3 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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.

3.5.4 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')
```

3.5.5 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 5.2 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 5.0 versions.

3.5.6 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.

3.5.7 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 4.11.80 : 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
```

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.

3.5.8 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
```

3.5.9 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 two new tables, `marketplace_pool` and `marketplaceapp_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
replicate-do-table  = opennebula.marketplace_pool
replicate-do-table  = opennebula.marketplaceapp_pool

# 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
```

3.5.10 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
```

3.5.11 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.

3.5.12 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"
```

3.5.13 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.5.14 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.

3.5.15 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.5.16 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');
```

3.6 Upgrading from OpenNebula 4.10.x

This section describes the installation procedure for systems that are already running a 4.10.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. The upgrade will preserve all current users, hosts, resources and configurations; for both Sqlite and MySQL backends.

Read the Compatibility Guide for 4.12, 4.14, 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

3.6.1 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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.6.2 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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.

3.6.3 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')
```

3.6.4 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 5.2 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 5.0 versions.

3.6.5 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.

3.6.6 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
```

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.

3.6.7 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
```

3.6.8 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 3 new tables, `vdc_pool`, `marketplace_pool` and `marketplaceapp_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
replicate-do-table  = opennebula.marketplace_pool
replicate-do-table  = opennebula.marketplaceapp_pool

# 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
```

3.6.9 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
```

3.6.10 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
```

3.6.11 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.

3.6.12 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"
```

3.6.13 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.

3.6.14 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 5.2.

3.6.15 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.6.16 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.

3.6.17 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.6.18 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');
```

3.7 Upgrading from OpenNebula 4.8.x

This section describes the installation procedure for systems that are already running a 4.8.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and *5.2*, and the Release Notes to know what is new in OpenNebula 5.2.

3.7.1 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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.7.2 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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
```

3.7.3 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.YYYY-MM-DD
```

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

3.7.4 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 5.2 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 5.0 versions.

3.7.5 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.

3.7.6 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.

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
```

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.

3.7.7 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
```

3.7.8 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 3 new tables, `vdc_pool`, `marketplace_pool` and `marketplaceapp_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
replicate-do-table   = opennebula.marketplace_pool
replicate-do-table   = opennebula.marketplaceapp_pool

# 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
```

3.7.9 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.

3.7.10 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"
```

3.7.11 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 5.2.

3.7.12 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.7.13 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.

3.7.14 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.7.15 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');
```

3.8 Upgrading from OpenNebula 4.6.x

This section describes the installation procedure for systems that are already running a 4.6.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

3.8.1 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 section for the **master zone**. After the master has been updated to 5.2, upgrade each **slave zone** following this same section.

3.8.2 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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
```

3.8.3 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.YYYY-MM-DD
```

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

3.8.4 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 5.2 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 5.0 versions.

3.8.5 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.

3.8.6 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.

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
```

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.

3.8.7 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
```

3.8.8 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 3 new tables, `vdc_pool`, `marketplace_pool` and `marketplaceapp_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
replicate-do-table  = opennebula.marketplace_pool
replicate-do-table  = opennebula.marketplaceapp_pool

# 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
```

3.8.9 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.

3.8.10 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 5.2.

3.8.11 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.8.12 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.

3.8.13 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.8.14 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');
```

3.9 Upgrading from OpenNebula 4.4.x

This section describes the installation procedure for systems that are already running a 4.4.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and *5.2*, and the Release Notes to know what is new in OpenNebula 5.2.

3.9.1 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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
```

3.9.2 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.

3.9.3 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 5.2 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 5.0 versions.

3.9.4 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
```

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.

3.9.5 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
```

3.9.6 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.

3.9.7 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 *"
```

3.9.8 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.9.9 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.

3.9.10 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.

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

3.9.11 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');
```

3.10 Upgrading from OpenNebula 4.2

This section describes the installation procedure for systems that are already running a 4.2 OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

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.

3.10.1 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- 802.1Q
- dummy
- ebtables
- fw
- ovswitch
- vxlan

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
```

3.10.2 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.

3.10.3 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 5.2 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 5.0 versions.

3.10.4 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.

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.

3.10.5 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
```

3.10.6 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.

3.10.7 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 *"
```

3.10.8 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.10.9 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.

3.10.10 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.

3.10.11 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.10.12 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');
```

3.10.13 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"
]
```

3.11 Upgrading from OpenNebula 4.0.x

This section describes the installation procedure for systems that are already running a 4.0.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and *5.2*, and the Release Notes to know what is new in OpenNebula 5.2.

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.

3.11.1 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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
```

3.11.2 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.

3.11.3 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 5.2 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 5.0 versions.

3.11.4 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
```

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

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.

3.11.5 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
```

3.11.6 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.

3.11.7 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.

3.11.8 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 *"
```

3.11.9 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.11.10 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.

3.11.11 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.11.12 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');
```

3.11.13 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"
]
```

3.12 Upgrading from OpenNebula 3.8.x

This section describes the installation procedure for systems that are already running a 3.8.x OpenNebula. The upgrade to OpenNebula 5.2 can be done directly following this section, you don't need to perform intermediate version upgrades. 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, 4.14, 5.0 and 5.2, and the Release Notes to know what is new in OpenNebula 5.2.

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.

3.12.1 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.

The network drivers since OpenNebula 5.0 are located in the Virtual Network, rather than in the host. The upgrade process may ask you questions about your existing VMs, Virtual Networks and hosts, and as such it is wise to have the following information saved beforehand, since in the upgrade process OpenNebula will be stopped.

```
$ onevnet list -x > networks.txt
$ onehost list -x > hosts.txt
$ onevm list -x > vms.txt
```

The list of valid network drivers since 5.0 Wizard are:

- `802.1Q`
- `dummy`
- `ebtables`
- `fw`
- `ovswitch`
- `vxlan`

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
```

3.12.2 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.

3.12.3 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 5.2 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 5.0 versions.

3.12.4 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 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
```

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.

3.12.5 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
```

3.12.6 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
```

3.12.7 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" ]
```

3.12.8 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"
]
```

3.12.9 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.

3.12.10 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.

3.12.11 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 *"
```

3.12.12 Create the Virtual Router ACL Rule

There is a new kind of resource introduced in 5.0: Virtual Routers. If you want your existing users to be able to create their own Virtual Routers, create the following ACL Rule:

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

Note: For environments in a Federation: This command needs to be executed only once in the master zone, after it is upgraded to 5.2.

3.12.13 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.

3.12.14 Restoring the Previous Version

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

- With OpenNebula 5.2 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.2, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.12.15 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');
```