OpenNebula 5.9 Introduction and Release Notes

Release 5.9.90

OpenNebula Systems

Nov 11, 2019

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

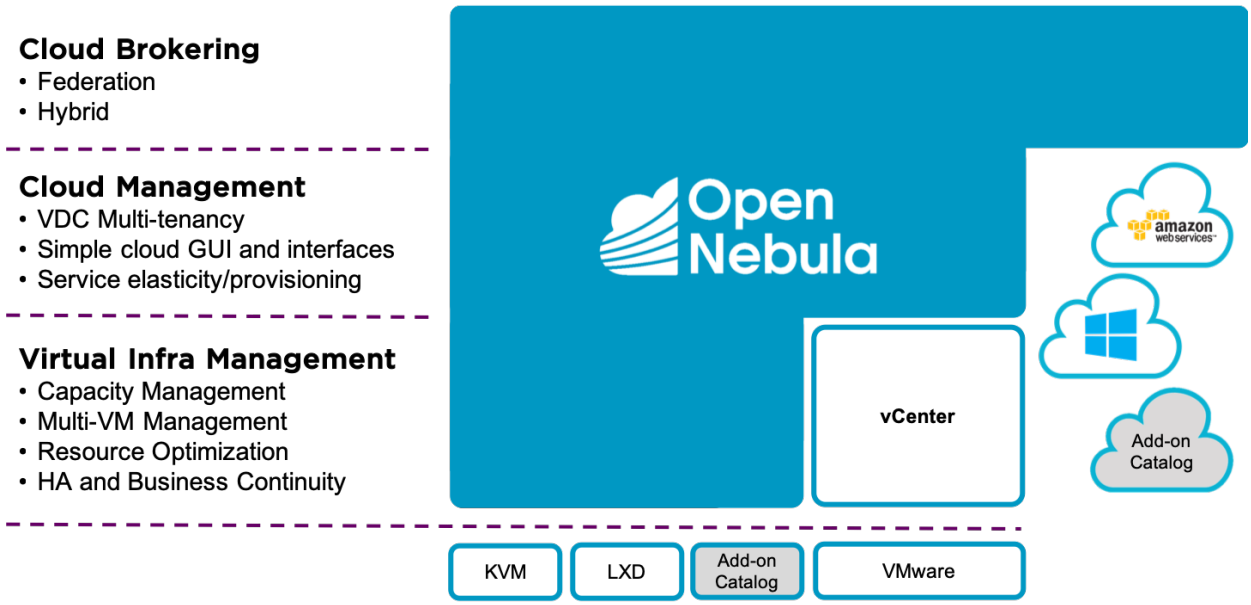
The first step is to decide on the hypervisor that you will use in your cloud infrastructure. The main OpenNebula distribution provides full support for the two most widely used hypervisors, KVM and VMware (through vCenter), at different levels of functionality.

- **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.
- **Containerization with LXD**. Containers are the next step towards virtualization. They have a minimal memory footprint and skip the compute intensive and sometimes unacceptable performance degradation inherent to hardware emulation. You can have a very high density of containers per virtualization node and run workloads close to bare-metal metrics. LXD focuses con system containers, instead of similar technolgies like Docker, which focuses on application containers.

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 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 an 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 LXD 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, LXD 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 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 or LXD, 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, LXD 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, an 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 and LXD 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 the 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.

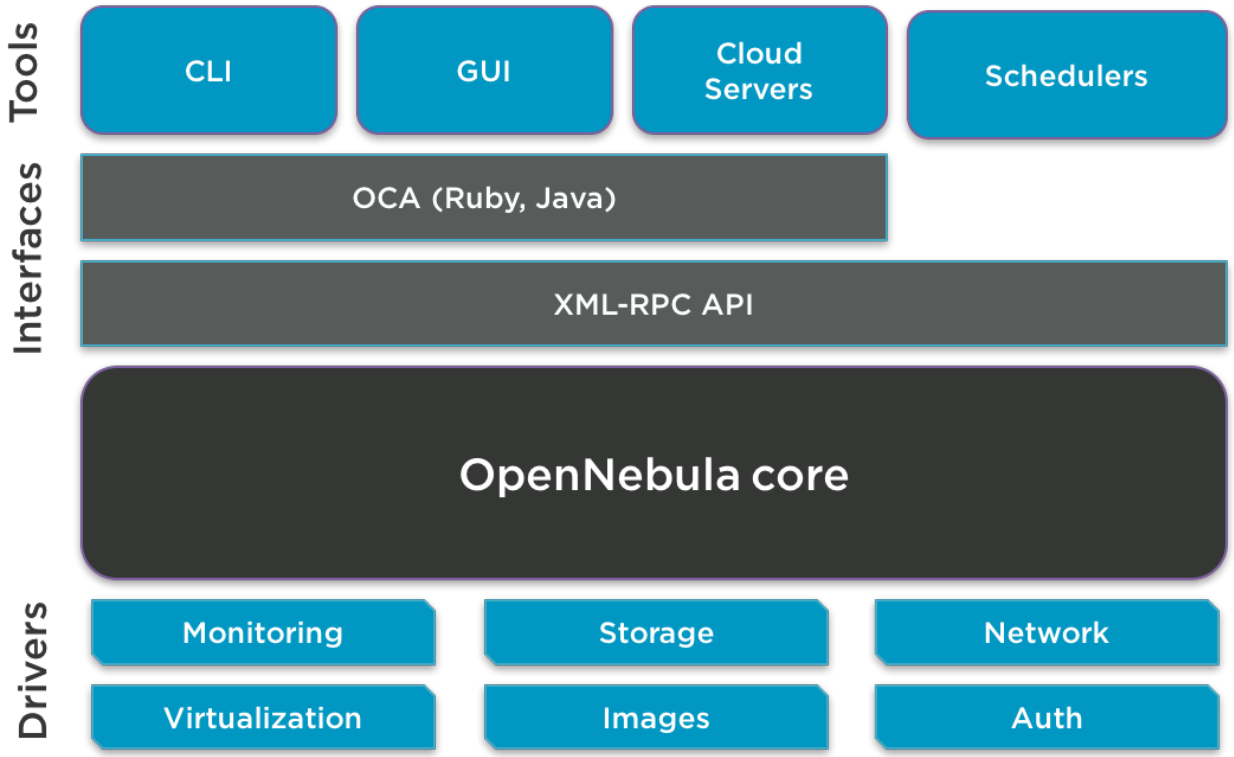
- **Cloud bursting** gives support to build a hybrid cloud, an extension of a private cloud to combine local resources with resources from remote cloud providers. A whole public cloud provider can be encapsulated as a local resource to be able to use extra computational capacity to satisfy peak demands. Out of the box connectors are shipped to support Amazon EC2 and Microsoft Azure cloudbursting.
- Disaggregated Data Centers provides tools to build and grow the physical infrastructure with resources from public bare-metal cloud providers, e.g. Amazon EC2 and Packet. New physical machines are allocated from the provider, configured to run the hypervisor and added into the OpenNebula as new clusters with hosts, datastores and virtual networks.

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, virtualization, 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 AND CONTAINER 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 and network hotplugging
- Disk snapshot capabilities and disk resizing for KVM and LXD instances
- LXD Containers are treated the same way as VMs in OpenNebula and support most of the VM features.

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 and LXD instances

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 access 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
- Advanced access control policies for VMs to redefine the access level (ADMIN, MANAGE and USE) required for each VM action
- Traceability on VM actions, VM history records logs the data associated to the action performed on a VM

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...
- Definition of groups of related VMs and set VM affinity rules across them.

- 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/LXD instance 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, LXD 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

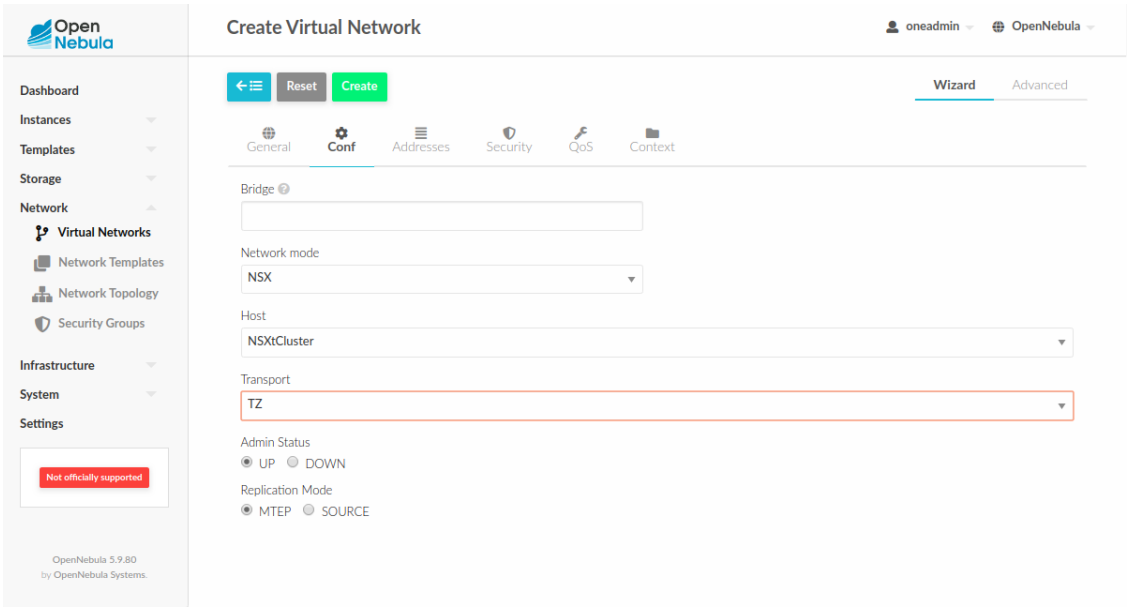
RELEASE NOTES 5.9.90

2.1 What's New in 5.10

This is the release candidate (5.9.90) release of OpenNebula 5.10

OpenNebula 5.10 (Boomerang) is the sixth major release of the OpenNebula 5 series. Main focus have been to enforce functionality to manage NFVs (as well as other workloads) to impulse OpenNebula as the default orchestrator of choice to build clouds in the edge and in environments where network performance is key. Also this focus on networking explains the new NSX integration over VMware infrastructures, which enables very interesting use cases in vSphere. The highlights of Boomerang are:

- **NUMA and CPU pinning**, define in which NUMA node VMs are going to be deployed.
- **NSX integration**, create and consume NSX networks from within OpenNebula.
- **Revamped hook subsystem**, hook a script for any API call or change of state in any VM or host resource.
- **DPDK support**, dramatically increase performance in network hungry, densely packed VMs.
- **2FA Authentication** for Sunstone.



As usual, OpenNebula 5.10 codename refers to a nebula, in this case the Boomerang Nebula, a protoplanetary nebula located 5,000 light-years away from Earth in the constellation Centaurus. It is also known as the Bow Tie Nebula and catalogued as LEDA 3074547. The nebula's temperature is measured at 1 K (-272.15 °C; -457.87 °F) making it the coolest natural place currently known in the Universe. Same as OpenNebula in the IaaS space :)

The OpenNebula team is now transitioning to "bug-fixing mode". Note that this is a 5.10 release candidate aimed at testers and developers to try the new features, and send a more than welcomed feedback for the final release. Also note that there is no migration path from the previous stable version (5.8.5) nor migration path to the final stable version (5.10.0). A list of open issues can be found in the GitHub development portal.

In the following list you can check the highlights of OpenNebula 5.10 (a detailed list of changes can be found here):

2.1.1 OpenNebula Core

- **Update hashing algorithm**, now passwords and login tokens are hashed using sha256 instead of sha1. Also csrftoken is now hashed with SHA256 instead of MD5
- **NUMA and CPU pinning**, you can define virtual NUMA topologies and pin them to specific hypervisor resources. NUMA and pinning is an important feature to improve the performance of specific workloads. You can read more here.
- **Live update of context information**, running VMs can update its context information and trigger the contextualization scripts in the guests, see here.
- **Uniform thread-safe random generator**, for random numbers use Mersenne Twister generator with uniform distribution.
- **VM operations configurable at user and group level**, use attributes `VM_USE_OPERATIONS`, `VM_MANAGE_OPERATIONS` and `VM_ADMIN_OPERATIONS` in user or group template, more information
- **Unified objects' secrets handling**, secrets are encrypted and decrypted in core, drivers get secrets decrypted see here.
- **Allow VM reschedule in poweroff state**, see here.
- **System wide CPU model configuration**, default cpu model for kvm could be set in config file see here.
- **KVM configuration per Host or Cluster**, all kvm default attributes can be overriden in Cluster and Host.
- **Revamped Hook System**, a more flexible and powerful hook system has been developed for 5.10. Now you can hook on any API call as well as state changes

Other minor features in OpenNebula core:

- FILTER is now a VM_RESTRICTED attribute.
- Increase size of indexes (log_index and fed_index) of logdb table from int to uint64.

Storage

- **Custom block size for Datablocks**, to allow users to modify block size for dd commands used for Ceph, Fs and LVM datastore drivers.
- **Configurable VM monitoring**, you can configure the frequency to monitor VM disk usage in datastores drivers (Fs and LVM). Check the oned.conf reference guide.
- **Extensible mixed modes**, different TM drivers can be easily combined by implementing custom driver actions for any combination. Check the storage integration guide for more details.
- **Support for Trash in Ceph datastore**, Allows user to send disks to the trash instead of removing them.

Networking

- **DPDK Support**, the Open vSwitch drivers include an option to support DPDK datapaths, read more here.
- **Extensible Network Drivers**, You can extend network driver actions with customizable hooks, see more details.
- **Deprecate brctl**, ip-route2 toolset replaces brctl to manage bridges for the KVM/LXD networking.

Sunstone

- **Two Factor Authentication**, with this method, not only does it request a username and password, it also requires a token generated by any of these applications: Google Authentication, Authy or Microsoft Authentication. You can read more here.

2.1.2 vCenter

- All VMM driver actions receive relevant information through stdin, saving oned calls and thus enhancing performance.
- The possibility to change the port used when OpenNebula connects to vSphere's API.
- Fixes an issue that causes missing parameters when import a vcenter network.
- Fixes an issue that causes VMware VM import to fail when it has disks with the same name in multiple datastores.

2.1.3 OneFlow & OneGate

- **Remove attributes from VMs**, the onegate server API supports a new option to delete attributes from VM user template via onegate command.

2.1.4 CLI

- **Better output for CLI tools**, new options to adjust and expand the output to the terminal size, also it allow better parsing of output, check the documentation (expand, adjust and size attributes) for more details.
- **Show raw ACL string in oneacl**, the full string of each rule can be shown. It's disabled by default check oneacl for more information.
- **Show orphan images** by using `oneimage orphans` commands.
- **Show orphan vnets** by using `onevnet orphans` commands.

2.1.5 Packaging

- **Packaged all required Ruby gems**, installation is now done only from operating system packages and `install_gems` is not necessary to run after each installation or upgrade anymore, check the front-end installation.
- Debian and Ubuntu debug packages, debugging information for the OpenNebula server are now dedicated package **opennebula-dbgsym**.
- Build optimizations, packages build respects the proposed compiler and linker parameters of each platform with additional hardening features.

- Node packages revert changes on uninstall, configuration changes in libvirt made during the KVM node package install. are reverted on uninstall.
- Avoid node_modules files in Sunstone package, built-time only data were dropped from distribution package.
- Sunstone package should not provide empty /var/lib/one/sunstone/main.js, temporary file with initially empty content is not contained in the package, but created by post-install scripts.
- Datastores directories contained in the package, initial datastores directories are not contained in the package anymore.
- Lower services restart interval, decreases limit for automatic restart of core services and consistently sets automatic restart to all services.
- Augeas lens for oned.conf, server package contains Augeas lens to manipulate `oned.conf`-like files.
- Optional Python bindings are now build also for Python 3 – package python3-pyone.
- Reviewed sudo-enabled commands, obsolete sudo-enabled commands were removed and rest commands are now enabled by each installed OpenNebula component package (server, node KVM, node LXD) to provide more fine-grained security.
- Packaged files and directories have more restricted ownership and permissions across all platforms, see here.

2.1.6 IPAM Drivers

- IPAM driver scripts now recieve the template of the AR via STDIN instead of via arguments, see more details.

2.1.7 KVM Monitoring Drivers

- KVM monitor scripts return host CPU model.

2.1.8 KVM Virtualization Driver

- A new option to sync time in guests has been added, see more details.

2.1.9 Other Issues Solved

- Fixes an issue that makes the network drivers fail when a large number of secturiy groups rules are used.
- Remove resource reference from VDC when resource is erased.
- Validate disk-snapshot-id cli parameter to prevent confusing conversion.
- Fix *Argument list too long* error in migrate action.
- Fix cluster CPU/MEM reservations.
- Fix issue with wrong controller for multiple scsi disks.
- Fix issue with Context ISO device vs. KVM models.
- Fix delete IPAM address ranges when deleting the vnet.
- Fix multiple click to back button when instantiate multiple VM.
- Fix add and remove cluster in datastore's table.
- Fix remove resource from VDC.

- Fix empty scheduled action id when is 0.
- Change order columns in services instances view.
- Fix send requeriments when a template is instantiated in user view.
- Fix lose NIC index in VM networks.
- Fix sunstone submit context in Virtual Network Template form.
- Fix FILES_DS template variable disappears if the configuration is updated.
- Fix wrong running quotas values when disk-snapshot create.
- Fix escape of backslash in XML documents for the onedb command.
- Add migrate power off in sunstone view yamls files.
- Fix preserve attributes in Virtual Machine Template.
- Fix libvirt race condition when detaching network interface.
- Fix hide the create button when it not have options.
- Fix parse error in VM descriptions with spaces.
- Fix error on resize VM disk in Firefox.
- Fix only show update if the version is stable.
- Fix update CPU model in VM config view.
- Fix showing uplinks as networks in vcenter hosts.
- Add the possibility of exclude some addresses from the HTTP proxy.
- Improve performance for large fileset containers.
- Fix show error when disable OpenNebula Systems support endpoint.

2.2 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.

2.2.1 Certified Components Version

Front-End Components

Component	Version	More information
RedHat Enterprise Linux	7, 8	Front-End Installation
CentOS	7, 8	Front-End Installation
Ubuntu Server	16.04 (LTS), 18.04 (LTS), 19.04, 19.10	Front-End Installation
Debian	9, 10	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 or 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	6.0/6.5/6,7, managing ESX 6.0/6.5/6.7	vCenter Node Installation
NSX-T	2.4.1+	VMware compatiblity. TODO NSX DOC
NSX-V	6.4.5+	VMware compatiblity. TODO NSX DOC

KVM Nodes

Component	Version	More information
RedHat Enterprise Linux	7, 8	KVM Driver
CentOS	7, 8	KVM Driver
Ubuntu Server	16.04 (LTS), 18.04 (LTS), 19.04, 19.10	KVM Driver
Debian	9, 10	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

LXD Nodes

Component	Version	More information
Ubuntu Server	16.04 (LTS), 18.04 (LTS), 19.04, 19.10	LXD Driver
Debian	10	LXD Driver
LXD	Support for LXD = 3.0.x either snap or system package	LXD Node Installation

Linux Contextualization Packages

Component	Version	More information
Amazon Linux	2	Linux Contextualization Packages
CentOS	6, 7, 8	Linux Contextualization Packages
Red Hat Enterprise Linux	7, 8	Linux Contextualization Packages
Fedora	29, 30	Linux Contextualization Packages
openSUSE	15, Tumbleweed	Linux Contextualization Packages
SUSE Linux Enterprise Server	12 SP3	Linux Contextualization Packages
Debian	8, 9, 10	Linux Contextualization Packages
Devuan	2	Linux Contextualization Packages
Ubuntu	14.04, 16.04, 18.04, 19.04	Linux Contextualization Packages
Alpine Linux	3.8, 3.9, 3.10	Linux Contextualization Packages
FreeBSD	11.3, 12.0	Linux Contextualization Packages

Windows Contextualization Packages

Component	Version	More information
Windows	7+	Windows Contextualization Packages
Windows Server	2008+	Windows Contextualization Packages

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	Jewel v10.2.x, Luminous v12.2.x, Mimic v13.2.x, Nautilus v14.2.x	The Ceph Datastore

Authentication

Component	Version	More information
net-ldap ruby library	0.12.1 or 0.16.1	LDAP Authentication
openssl	Version included in the Linux distribution	x509 Authentication

Cloud Bursting

Component	Version	More information
aws-sdk	2.11.330	Amazon EC2 Driver
azure	0.7.10	Azure Driver
one-to-one	1.0.0	OpenNebula Driver

Application Containerization

Component	Version
Docker	18.03.0-ce
Docker Machine	0.14.0
Appliance OS	Ubuntu 16.04

Sunstone

Browser	Version
Chrome	61.0 - 67.0
Firefox	59.0 - 61.0
IE	11.0

Note: For Windows desktops using **Chrome** or **Firefox** you should disable the option `touch-events` of your browser:

Chrome: chrome://flags -> #touch-events: disabled. **Firefox**: about:config -> dom.w3c_touch_events: disabled.

Internet Explorer is **not** supported with the Compatibility Mode enabled, since it emulates IE7 which is not supported.

VMware Cloud on AWS

OpenNebula has been validated and is supported on VMware Cloud on AWS. Customers can contact the support team through the commercial support portal to know specific configuration and limitations.

Note: Generally for all Linux platforms, it is worth noting that Ruby gems should be used from packages shipped with the OpenNebula or installed with the install_gems utility. Avoid using Ruby gems versions shipped with your platform.

2.2.2 Certified Infrastructure Scale

A single instance of OpenNebula (ie, a single `oned` process) has been stress tested to cope with 500 hypervisors without performance degradation. This is the maximum recommended configuration for a single instance, and depending on the underlying configuration of storage and networking mainly, it is recommended to switch to a federated scenario for any larger number of hypervisors.

However, there are several OpenNebula users managing significant higher number of hypervisors (in the order of two thousand) with a single instance, this largely depends as mentioned on the storage, networking and also monitoring configuration.

2.2.3 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.
- Only **Ruby versions >= 2.0 are supported**.

Ubuntu 16.04 Platform Notes

By default it comes with LXD 2, LXD 3 should be installed from **xenial-backports**. Make sure you have backports enabled in sources.list

```
# apt-get -t xenial-backports install lxd
```

Resizing **ext4** filesystems of LXD containers will fail due to outdated `e2fsck` package

CentOS 7.0 Platform Notes

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).

2.2.4 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 Platform Notes

Ruby Dependencies

In order to install Ruby dependencies on RHEL, 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"
...
```

CentOS/RedHat 8 Platform Notes

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 Compatibility Guide

This guide is aimed at OpenNebula 5.9.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 5.9.

2.3.1 Network Driver actions interface

The way arguments are passed to the pre/post/clean/update_sg has changed as follows:

- The old argument 1 `vm xml template` it is now sent through by `stdin`
- The old argument 2 `vm deploy id` now is argument 1
- There is no argument 2

This change has been introduced because of this bug.

2.3.2 Bridge Interface options

As `Bridge utils (brctl)` bacame obsolete they were replaced by `ip-route2`. Bridge options for `ip command` could be specified in `:ip_bridge_conf` but for backward compatibility the section `:bridge_conf` is still accepted and options are transformed to the `ip-route2` format.

2.3.3 Password Hashing Algorithm Update

User passwords and login tokens are now generated using SHA256 instead of SHA1. OpenNebula core will update users passwords in the database when they first login in the system. It is recommened to request your users to login after the upgrade.

2.3.4 Packages

OpenNebula now ships with distribution packages for all required Ruby gems, executing of the `install_gems` script after installation or upgrade is not necessary anymore. Ruby dependencies are installed into a dedicated directory `/usr/share/one/gems-dist/` and OpenNebula uses them exclusively via symlinked location `/usr/share/one/gems/`. **System-wide Ruby gems are not used anymore!** Any Ruby gems needed by the custom drivers need to be installed again into a new dedicated location. Check the details in Front-end Installation.

If Sunstone is running via Passenger in Apache, it might be necessary to set `GEMS_HOME` and `GEMS_PATH` environment variables to `/usr/share/one/gems/` to force the Ruby running inside the web server to use these new location. Check the details in Sunstone for Large Deployments.

2.3.5 IPAM Drivers

IPAM driver scripts now recieve the template of the AR via STDIN instead of via arguments.

2.3.6 OpenNebula Core

The `DEFAULT_DEVICE_PREFIX` configuration variable is now set to `sd` by default.

2.3.7 Hooks

Hooks have been redesign and you need to update your hook definition to the new system. *Please, follow the instructions in the upgrade guide to update your hooks*.

2.4 Known Issues

A complete list of known issues for OpenNebula is maintained here.

This page will be updated with relevant information about bugs affecting OpenNebula, as well as possible workarounds until a patch is oficially published.

2.4.1 NIC alias and IP spoofing rules

For NIC alias the IP spoofing rules are not triggered when the VM is created nor when the interface is attached. If you have configured IP spoofing for your virtual networks be aware that those will not be honored by NIC ALIAS interfaces. More info here.

2.4.2 CLI warning message

Using some CLI commands in Ubuntu 18.04, due to ruby and gems version, you may see this message:

warning: constant ::Fixnum is deprecated

As a workaround you can use *export RUBYOPT="-W0*, this will disable the warning message (but, take in account that it will disable all warning messages from ruby)

2.5 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.

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 5.8.x

This section describes the installation procedure for systems that are already running a 5.8.x OpenNebula. The upgrade to OpenNebula 5.9 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 5.9.

3.2.1 Upgrading Single Front-end Deployments

Step 1. Check Virtual Machine Status

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.

Step 2. Stop OpenNebula

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.

Step 3. Backup OpenNebula Configuration

Backup the configuration files located in **/etc/one** and **/var/lib/one/remotes/etc**. 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.$(date +'%Y-%m-%d')
# cp -r /var/lib/one/remotes/etc /var/lib/one/remotes/etc.$(date +'%Y-%m-%d')
```

Step 4. Upgrade to the New Version

Ubuntu/Debian

```
# apt-get install --only-udpate opennebula opennebula-sunstone opennebula-gate␣
↪opennebula-flow python-pyone
```

CentOS

```
# yum upgrade opennebula-server opennebula-sunstone opennebula-ruby opennebula-gate␣
↪opennebula-flow
```

Step 5. Update Configuration Files

If you haven't modified any configuration files, you can skip this step and procceed to the next one.

In order to update the configuration files with your existing customizations you'll need to:

1. Compare the old and new configuration files: `diff -ur /etc/one.YYYY-MM-DD /etc/one` and `diff -ur /var/lib/one/etc.YYYY-MM-DD /var/lib/one/etc`. You can use graphical diff-tools like `meld` to compare both directories, which are very useful in this step.
2. Edit the **new** files and port all the customizations from the previous version.

Step 6. Upgrade the Database version

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

Simply run the 'onedb upgrade -v' command. The connection parameters have to be supplied with the command line options, see the onedb manpage for more information. For example:

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

Step 7. Check DB Consistency

First, move the 5.9 backup file created by the upgrade command to a safe place. If you face any issues, the onedb command can restore this backup, but it won't downgrade databases to previous versions. Then execute the *onedb fsck* command providing the same connection parameter used during the database upgrade:

```
$ 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
```

Step 8. Start OpenNebula

Make the system to re-read the service configuration files of the new packages:

```
# systemctl daemon-reload
```

Now you should be able to start OpenNebula as usual, running `service opennebula start` as `root`. Do not forget to restart also any associated service like Sunstone, OneGate or OneFlow.

At this point 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. You may also try some **show** subcommand for some resources to check everything is working (e.g. `onehost show`, or `onevm show`).

Step 9. Update the Hypervisors (LXD & KVM only)

First update the virtualization, storage and networking drivers, as `oneadmin` user execute:

```
$ onehost sync
```

Then log into you hypervisor hosts and update the opennebula-node packages:

Ubuntu/Debian

```
# apt-get install --only-upgrade opennebula-node
# service libvirtd restart # debian
# service libvirt-bin restart # ubuntu
```

If upgrading the LXD drivers on Ubuntu

```
# apt-get install --only-upgrade opennebula-node-lxd
```

CentOS

```
# yum upgrade opennebula-node-kvm
# systemctl restart libvirtd
```

3.2.2 Upgrading High Availability Clusters

Step 1. Stop the HA Cluster

You need to stop all the nodes in the cluster to upgrade them at the same time. Start from the followers and leave the leader to the end.

Step 2. Upgrade the Leader

Follow Steps 3 to 7 described in the previous Section (Upgrading Single Front-end deployments). Finally create a database backup to replicate the *upgraded* state to the followers:

```
$ onedb backup -u oneadmin -p oneadmin -d opennebula
MySQL dump stored in /var/lib/one/mysql_localhost_opennebula_2019-9-27_11:52:47.sql
Use 'onedb restore' or restore the DB using the mysql command:
mysql -u user -h server -P port db_name < backup_file
```

Step 3. Upgrade OpenNebula in the Followers

Upgrade OpenNebula packages as described in Step 4 in the previous section (Upgrading Single Front-end deployments)

Step 4. Replicate Database and configuration

Copy the database backup of the leader to each follower and restore it:

```
$ scp /var/lib/one/mysql_localhost_opennebula_2019-9-27_11:52:47.sql <follower_ip>:/
↪tmp

$ onedb restore -f -u oneadmin -p oneadmin -d opennebula /tmp/mysql_localhost_
↪opennebula_2019-9-27_11:52:47.sql
MySQL DB opennebula at localhost restored.
```

Synchronize the configuration files to the followers:

```
$ rsync -r /etc/one root@<follower_ip>:/etc

$ rsync -r /var/lib/one/remotes/etc root@<follower_ip>:/var/lib/one/remotes
```

Step 5. Start OpenNebula in the Leader and Followers

Start OpenNebula in the followers as described in Step 8 in the previous section (Upgrading Single Front-end deployments).

Step 6. Check Cluster Health

At this point `onezone show` command should render all the followers active and in sync with the leader.

Step 7. Update the Hypervisors (KVM & LXD)

Finally upgrade the hypervisors as described in Step 9 in the previous section (Upgrading Single Front-end deployments).

3.2.3 Upgrading a Federation

This version of OpenNebula does not upgrade the shared database schema. The federation can be upgraded zone by zone. For each zone please follow the previous procedure that applies to your setup.

3.2.4 Update your Hooks

Hooks are no longer defined in `oned.conf`, you need to recreate any hook you are using in the OpenNebula database. Specific upgrade actions for each hook type are described below.

RAFT/HA Hooks

HA Hooks keep working as they did in previous versions. For design reasons, these are the only hooks which need to be define in oned.conf and cannot be managed via API or CLI. You should preserve your previsous configuration in oned.conf.

Fault Tolerance Hooks

In order to migrate fault tolerance hooks just follow the steps defined in Fault Tolerance guide.

vCenter Hooks

The vCenter Hooks, used for creating virtual networks, will be created automatically when needed.

Custom Hooks

Custom Hooks migration strongly depend on the use case trying to solve with the hook. Below there is a list of examples which represent the most common use cases.

- Create/Remove hooks. Corresponds to the legacy `ON=CREATE` and `ON=REMOVE` hooks

These hooks are now triggered by an API hook on the corresponding create/delete API call. For example, the following hook sends an email to the user when her user account is created:

```
USER_HOOK = [
    name      = "mail",
    on        = "CREATE",
    command   = "email2user.rb",
    arguments = "$ID $TEMPLATE"]
```

Now, in OpenNebula 5.10, you need to create the following hook template:

```
NAME      = "mail",
TYPE      = API
CALL      = "one.user.allocate",
COMMAND   = "email2user.rb",
ARGUMENTS = "$TEMPLATE"
```

and define the hook with `onehook create` command.

In general, any create/remove hook can be migrated using the following template:

```
NAME = hook-create-resource
TYPE = api
COMMAND = "<same-script-path>"
ARGUMENTS = "<same-arguments>"
CALL = "one.<resource>.allocate"
```

More information on defining API Hooks can be found here.

- State hooks

If there is a hook defined for a Host or VM state change, the hook template have to be inferred from the Hook definition in the 5.8 `oned.conf` file, see the example below:

```
# Legacy hook definition in oned.conf

    VM_HOOK = [
    name       = "advanced_hook",
    on         = "CUSTOM",
    state      = "ACTIVE",
    lcm_state  = "BOOT_UNKNOWN",
    command    = "log.rb",
    arguments = "$ID $PREV_STATE $PREV_LCM_STATE" ]

# Hook template file

    NAME = advanced_hook
    TYPE = state
    COMMAND = "log.rb"
    ARGUMENTS = "$TEMPLATE"
    RESOURCE = VM
    ON = CUSTOM
    STATE = ACTIVE
    LCM_STATE = BOOT_UNKNOWN
```

Note that you may need to adapt the arguments of your hook, as `$ID` is not currently supported. More information on defining state Hooks can be found here.

Note: Note that, in both examples, `ARGUMENTS_STDIN=yes` can be used for passing the parameters via STDIN instead of command line argument.

3.2.5 Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.3 Upgrading from OpenNebula 5.6.x

This section describes the installation procedure for systems that are already running a 5.6.x OpenNebula. The upgrade to OpenNebula 5.9 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 5.9.

3.3.1 Upgrading a Federation and High Availability

You need to perform the following steps in all the HA nodes and all zones. Note that you need to update all the servers at the same time, not one by one.

3.3.2 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.

Stop OpenNebula

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.

3.3.3 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')
```

3.3.4 Installation of New Version

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 5.9 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` 5.9 reference.

3.3.5 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.

Note: Configuration files from inside the remote scripts directory structure `/var/lib/one/remotes/` have been moved into dedicated directory `/var/lib/one/remotes/etc/`. Check all the files on the new path, and apply any necessary changes to your environment.

3.3.6 Database Upgrade

Perform the 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.

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

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

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

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

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

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

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


>>> Running migrators for shared tables
  ...

>>> Running migrators for local tables
```

(continues on next page)

(continued from previous page)

```
  ...
  > 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.

Note: vCenter VM disks managed by OpenNebula will be retagged in the vCenter VMs extraConfig. It is important that the front-end has access to the vCenter servers managed by OpenNebula in this DB upgrade process.

3.3.7 Check DB Consistency

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

First, move the 5.6.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
```

3.3.8 Reload Start Scripts

Follow this section if you are using a *systemd* base distribution, like CentOS 7+, Ubuntu 16.04+, etc.

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
```

3.3.9 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.

Note: You can skip this step if you are not using KVM hosts, or any hosts that use remove monitoring probes.

3.3.10 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.

3.3.11 Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.3.12 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');
```

3.3.13 Bug recovering

If Ceph datastores were used with OpenNebula <= 5.6.2 and any VM have been reverted to a snapshot, it's needed to follow the next steps for recovering snapshot tree consistency:

Warning: Check history in order to find how many reverts have been done. If the number of reverts are greater than 1 we do not recommend to deleted any snapshot, becase it will cause lose of information. If the number of revert is 1 you can fix it by following the steps below.

- Use the command `onedb update-body vm --id <vm_id>` for updating the body of the VM.
- Set /VM/SNAPSHOTS/CURRENT_BASE to the ID of the current active snapshot.

3.4 Upgrading from OpenNebula 5.4.x

This section describes the installation procedure for systems that are already running a 5.4.x OpenNebula. The upgrade to OpenNebula 5.9 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 5.9.

Warning: OpenNebula 5.4.1 modifies the existing Sunstone views configuration files ('/etc/one/sunstone-views/') to adjust the column names. Any change made in these files will need to be reapplied after the OpenNebula upgrade.

3.4.1 Upgrading a Federation and High Availability

You need to perform the following steps in all the HA nodes and all zones. Note that you need to update all the servers at the same time, not one by one.

3.4.2 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.

Stop OpenNebula

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.

3.4.3 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')
```

3.4.4 Installation of New Version

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 5.9 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` 5.9 reference.

3.4.5 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.

Important: Please adjust the XMLRPC_TIMEOUT according to the typical RTT (round-trip-time) of xml-rpc calls across RAFT servers. This value should be 4 or 5 times the average RTT

3.4.6 Database Upgrade

Perform the 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.

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

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

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

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

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

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

MySQL dump stored in /var/lib/one/mysql_localhost_opennebula.sql
Use 'onedb restore' or restore the DB using the mysql command:
```

(continues on next page)

(continued from previous page)

```
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.

3.4.7 Check DB Consistency

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

First, move the 5.4.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
```

3.4.8 Reload Start Scripts

Follow this section if you are using a *systemd* base distribution, like CentOS 7+, Ubuntu 16.04+, etc.

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
```

3.4.9 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.

3.4.10 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.

3.4.11 Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.4.12 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');
```

3.5 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 to OpenNebula 5.9 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 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to the latest version of OpenNebula.

3.5.1 Upgrading a Federation

If you have two or more 5.2.x OpenNebulas working as a Federation, you need to upgrade all of them. The upgrade for this version has to occur simultaneously on all zones including the master.

The steps to follow are:

1. Stop the MySQL replication in all the slaves and master zone. The MySQL replication is no longer needed.
2. Upgrade the **master** zone to the latest version
3. Upgrade the **slaves** zones to the latest version
4. Replicate the state of the shared tables from the master zone into each slave zone.

During steps 1 and 2 the slave OpenNebula's can be running, and users can keep accessing them if each zone has a local Sunstone instance. However all the shared database tables (users, groups, ACL...) will not be updated in the slaves zones till step 3 is completed.

To perform the first step, you must stop and reset each slave (and master), remove any configuration attribute for replication in `my.cnf` file and finally restart `mysqld`. Please refer to mysql documentation for more details on how to perform this step.

Then follow this section for the **master zone**. After the master has been updated to 5.9, upgrade each **slave zone** following this same section.

3.5.2 Upgrading from a High Availability deployment

You need to restore the HA deployment according to the new implementation. Upgrade the active OpenNebula instance as described in this section and then regenerate the HA instances as described in the in the HA guide.

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.

vCenter

Important: Read this section carefully if you are using vCenter!

If you are using vCenter you will need to follow some extra steps while **still running OpenNebula 5.2**.

Follow the *vCenter upgrade 5.2 to 5.4 Pre-migration phase*.

Stop OpenNebula

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 of New Version

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 5.9 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 5.4 and 5.9 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.

EC2 Configuration File

The credentials and capacity from ec2 zones have been moved from its configuration file to the template of the host. You don't need to update the file `/etc/one/ec2_driver.conf` with the data from the old file. To make this data available to the migrator copy the old configuration file to `/etc/one/ec2_driver.conf.old`:

```
# cp /etc/one.$(date +'%Y-%m-%d')/ec2_driver.conf /etc/one/ec2_driver.conf.old
```

After migration you can delete the old file:

```
# rm /etc/one/ec2_driver.conf.old
```

3.5.7 Database Upgrade

vCenter Migration Tool

Important: Read this section carefully if you are using vCenter!

If you are using vCenter you will need to run the vCenter migration tool before running the *onedb upgrade* command from the next section.

Follow the *vCenter upgrade 5.2 to 5.4 Migration phase*.

Perform the 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.

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

Note: 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
```

(continues on next page)

(continued from previous page)

```
  ...
  > 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 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
```

3.5.9 Recreate the Federation salves

This section applies only to environments working in a Federation.

For the **master zone**: Snapshot the shared tables using the `onedb` tool. Please refer to the federation guide for more details.

For a **slave zone**: Each slave should be already configured, i.e. `oned.conf` should include the `ZONE_ID` for the slave, auth files present and OpenNebula updated to last version. You only need to restore the shared tables saved in the previous step and start the slave zone.

3.5.10 Reload Start Scripts

Follow this section if you are using a *systemd* base distribution, like CentOS 7+, Ubuntu 16.04+, etc.

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 `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.

3.5.12 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.

3.5.13 Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.5.14 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');
```

3.6 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 5.9 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 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

3.6.1 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 5.9, upgrade each **slave zone** following this same section.

3.6.2 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.

3.6.3 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.

3.6.4 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')
```

3.6.5 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 5.9 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 5.9 versions.

3.6.6 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.

3.6.7 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.

3.6.8 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
```

3.6.9 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
```

3.6.10 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
```

3.6.11 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.

3.6.12 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.

3.6.13 Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

3.6.14 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');
```

3.7 vCenter upgrade 5.2 to 5.4

3.7.1 Pre-migration phase

OpenNebula provides a script that must be run **before** it is upgraded using the **oneadmin** user account. This script can be downloaded from https://downloads.opennebula.org/packages/opennebula-5.4.1/vcenter_one54_pre.rb.

Warning: As in 5.2 OpenNebula disks cannot have spaces in the VMDK paths. However, OpenNebula 5.4 now exposes all disks of existing templates and VMs. These disks were transparent for 5.2 and cannot have spaces so you need to remove them prior to upgrade. This limitation will be addressed in the short-term in the next maintenance release.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

Warning: The pre-migration phase may fail if there are resources in error, please clean resources in failed state prior to continue with this process.

The script will perform the following tasks:

- Establish a connection to every vCenter instance known by OpenNebula
- Retrieve information about clusters, virtual machines, templates, datastores and port groups.
- New information will be added to the OpenNebula resources.
- Some manual intervention may be required.
- For each IMAGE datastore found, a SYSTEM datastore will be created.
- Templates and wild VMs will be inspected in order to discover virtual hard disks and network interface cards that are invisible.
- All Datastores that hosts those virtual hard disks will be imported into OpenNebula.
- OpenNebula images and virtual networks will be created so the invisible disks and nics bebcome visible after upgrade.
- The virtual networks that represent port groups found inside existing templates will have an Ethernet address range with 255 MACs in the pool. You may want to change or increase this address range after the pre-migrator tool finishes.
- OpenNebula hosts, networks and datastores will grouped into OpenNebula clusters. Each vCenter cluster will be assigned to an OpenNebula cluster.
- XML files will be generated under `/var/tmp` directory. They will be used in the migration phase.

Important: Read carefully the instructions of the Phase 0. It involves modifying `/etc/one/oned.conf` and `/var/lib/one/remotes/datastore/vcenter/rm` and restarting OpenNebula. **DON'T FORGET TO DO SO**.

```
================================================================================
 / _ \ _ __   ___ _ __ | \ | | ___| |__  _   _| | __ _
| | | | '_ \ / _ \ '_ \|  \| |/ _ \ '_ \| | | | |/ _` |
| |_| | |_) |  __/ | | | |\  |  __/ |_) | |_| | | (_| |
 \___/| .__/ \___|_| |_|_| \_|\___|_.__/ \__,_|_|\__,_|
      |_|
--------------------------------------------------------------------------------
  vCenter pre-migrator tool for OpenNebula 5.4 - Version: 1.0
================================================================================

================================================================================
 PHASE 0 - Before running the script please read the following notes
================================================================================

- Please check that you have set PERSISTENT_ONLY="NO" and REQUIRED_ATTRS=""
```

(continues on next page)

(continued from previous page)

```
  in you DS_MAD_CONF vcenter inside the /etc/one/oned.conf and that you have
  restarted your OpenNebula services to apply the new configuration before
  launching the script.

- Edit the file /var/lib/one/remotes/datastore/vcenter/rm and replace the
  following lines:

  vi_client.delete_virtual_disk(img_src,
                                ds_name)

  with the following lines:

  if drv_action["/DS_DRIVER_ACTION_DATA/IMAGE/TEMPLATE/VCENTER_IMPORTED"] != "YES"
       vi_client.delete_virtual_disk(img_src,ds_name)
  end

  in order to avoid that you accidentally remove a virtual hard disk from a template
  or wild VM when you delete an image.

- Note that this script may take some time to perform complex tasks so please be␣
↪patient.

- Although this scripts will do its best to be fully automated there may be situations
  where a manual intervention is needed, in that case a WARNING will be shown.

- The virtual networks that represent port groups found inside existing templates
  will have an Ethernet address range with 255 MACs in the pool. You may want to
  change or increase this address range after the pre-migrator tool finishes.

- It's advisable to disable the Sunstone user interface before launching this script
  in order to avoid that OpenNebula objects created by users while
  the script is running are not pre-migrated by the tool.

- This script can be executed as many times as you wish. It will update previous
  results and XML template will be always overwritten.

Don't forget to restart OpenNebula if you have made changes!

Do you want to continue? ([y]/n):
```

In short, you need to replace the following in `/etc/one/oned.conf`:

```
DS_MAD_CONF = [
-    NAME = "vcenter", REQUIRED_ATTRS = "VCENTER_CLUSTER", PERSISTENT_ONLY = "YES",
+    NAME = "vcenter", REQUIRED_ATTRS = "", PERSISTENT_ONLY = "NO",
    MARKETPLACE_ACTIONS = "export"
]
```

And the following change in `/var/lib/one/remotes/datastore/vcenter/rm`:

```
-vi_client.delete_virtual_disk(img_src,
-                              ds_name)
+if drv_action["/DS_DRIVER_ACTION_DATA/IMAGE/TEMPLATE/VCENTER_IMPORTED"] != "YES"
+    vi_client.delete_virtual_disk(img_src,ds_name)
+end
```

Note: It's advisable to disable the Sunstone user interface while the pre-migrator script is run in order to avoid that OpenNebula objects created by users while the script is run are not pre-migrated.

In order to execute the script you need to download from https://downloads.opennebula.org/packages/opennebula-5.4.1/vcenter_one54_pre.rb and run it manually **as oneadmin**.

```
$ curl -skLO https://downloads.opennebula.org/packages/opennebula-5.4.1/vcenter_one54_
↪pre.rb
$ ruby vcenter_one54_pre.rb
```

Note: If you want to execute this script more than once, please delete `/var/tmp/vcenter_one54` directory.

3.7.2 OpenNebula Upgrade

Important: Now you need to continue upgrading the software following the steps described in the upgrade guide.

Follow the *Upgrade OpenNebula software*.

3.7.3 Migration phase

Once OpenNebula packages have been upgraded, you need to execute the pre migration tool for vCenter.

Warning: The migration tool must be run **before** a onedb upgrade command is executed.

The migration tool is launched using the `onedb vcenter-one54` command, and it must be run from the same machine where the pre-migrator tool was executed as it requires some XML templates files stored in the `/var/tmp` directory.

```
$ onedb vcenter-one54 -v -u <dbuser> -p <dbpass> -d <dbname> -S <dbhost>
```

The migration tool will update some OpenNebula's database tables using the XML files that were created in the pre-migration phase. This is the list of affected tables:

- `template_pool`
- `vm_pool`
- `host_pool`
- `datastore_pool`
- `network_pool`
- `image_pool`

In the following sections you will need to execute `onedb fsck`. Note that you might get the following error: `[UNREPAIRED] VM XX has a lease from VNet XX, but it could not be matched to any AR`. This is expected for previously invisible NIC interfaces in VMs added in the pre-migration phase.

Continue the upgrade by moving on to the *next section*.

3.8 Upgrading from 4.x.x

3.8.1 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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 5.9, upgrade each **slave zone** following this same section.

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.

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.

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')
```

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 5.9 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 5.9 versions.

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.

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.

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
```

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
```

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
```

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.

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.2 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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 5.9, upgrade each **slave zone** following this same section.

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.

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.

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')
```

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 5.9 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.

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.

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
```

(continues on next page)

(continued from previous page)

```
>>> 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.

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
```

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
```

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
```

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.

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"
```

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.3 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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 5.9, upgrade each **slave zone** following this same section.

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.

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')
```

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 5.9 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.

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.

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.

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
```

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
```

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
```

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
```

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.

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"
```

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.

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 5.9.

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.4 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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 5.9, upgrade each **slave zone** following this same section.

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
```

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.

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 5.9 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.

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.

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
```

(continues on next page)

(continued from previous page)

```
>>> 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.

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
```

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
```

(continues on next page)

(continued from previous page)

```
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
```

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.

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"
```

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 5.9.

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.5 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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 5.9, upgrade each **slave zone** following this same section.

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
```

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.

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 5.9 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.

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.

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.

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
```

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
```

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.

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 5.9.

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.6 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

> **Warning:** If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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
```

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.

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 5.9 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.

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.

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
```

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.

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 *"
```

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

3.8.7 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

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: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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
```

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.

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 5.9 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.

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.

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
```

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.

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 *"
```

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 5.9.

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.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

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"
]
```

3.8.8 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

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.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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
```

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.

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 5.9 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.

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.

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
```

(continues on next page)

(continued from previous page)

```
Total errors found: 0
```

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.

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.

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 *"
```

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```

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
```

(continues on next page)

(continued from previous page)

```
#       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"
]
```

3.9 Upgrading

3.9.1 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 5.9 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 *5.9*, and the Release Notes to know what is new in OpenNebula 5.9.

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.

Warning: If you are using the vCenter drivers, there is a manual intervention required in order to upgrade to OpenNebula 5.4. Note that **upgrading from OpenNebula < 5.2 to OpenNebula >= 5.4 is NOT supported**. You need to upgrade first to OpenNebula 5.2, and then upgrade to OpenNebula 5.4.

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
```

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.

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 5.9 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.

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
```

(continues on next page)

(continued from previous page)

```
  > 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
```

(continues on next page)

(continued from previous page)

```
  > 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.

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
```

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
```

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 = [
```

(continues on next page)

(continued from previous page)

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

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"
]
```

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.

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.

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 *"
```

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 5.9.

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.

Restoring the Previous Version

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

- With OpenNebula 5.9 still installed, restore the DB backup using 'onedb restore -f'
- Uninstall OpenNebula 5.9, and install again your previous version.
- Copy back the backup of /etc/one you did to restore your configuration.

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');
```