# Table of Contents

- **Introduction** .......................................................... 3
  - 1.1 VMware vSphere .......................................................... 3
  - 1.2 OpenStack ............................................................... 3
  - 1.3 Using OpenStack with vSphere ........................................... 4
    - 1.3.1 OpenStack and vSphere: Conceptual Analogies ....................... 5
  - 1.4 The VMware OpenStack Virtual Appliance .................................. 5
- **2. Requirements** ........................................................ 6
  - 2.1 vSphere Requirements .................................................... 6
    - 2.1.1 vSphere Inventory: Single Datacenter .................................. 6
    - 2.1.2 Cluster: Automated VMware vSphere Storage DRS ....................... 6
    - 2.1.3 Storage: Shared ....................................................... 6
    - 2.1.4 Networking ........................................................... 7
      - 2.1.4.1 Port Groups and VLANs ........................................... 7
      - 2.1.4.2 ESXi Firewall .................................................... 7
      - 2.1.4.3 DHCP Server ...................................................... 8
- **3. Installation** .......................................................... 8
  - 3.1 Importing VOVA ............................................................ 8
  - 3.2 Configuring VOVA ........................................................ 8
  - 3.3 Starting VOVA ............................................................ 8
  - 3.4 Configuring the VMware vSphere Web Client Plug-in for OpenStack .......... 9
- **4. Managing OpenStack with the Horizon Web Interface** ......................... 12
  - 4.1 Logging In – Web ......................................................... 12
  - 4.2 Logging In – SSH/CLI ...................................................... 12
  - 4.3 Flavor of the Day .......................................................... 12
  - 4.4 Launching an Instance .................................................... 13
  - 4.5 Accessing the Console for an Instance ..................................... 14
  - 4.6 Managing Storage ........................................................ 16
    - 4.6.1 Adding Persistent Storage to an Instance ................................ 16
    - 4.6.2 Removing Persistent Storage from an Instance .......................... 18
    - 4.6.3 Adding the Persistent Storage to Another Instance ..................... 18
  - 4.7 Network ................................................................. 19
- **Summary** .................................................................. 20
- **Acknowledgments** ....................................................... 20
- **About the Authors** ....................................................... 20
Introduction

1.1 VMware vSphere

VMware vSphere® is the industry’s leading and most reliable virtualization and cloud computing platform. vSphere simplifies IT by separating applications and operating systems (OSs) from the underlying hardware. Existing applications receive dedicated resources, but servers can be centrally managed and monitored as a pool of shared resources that can be offered as a service to application infrastructure consumers.

Some believe that a choice must be made to use either VMware® products or OpenStack, but this is not the case. In fact, VMware compute, networking, and storage technologies bring unique capabilities to OpenStack cloud deployments.

1.2 OpenStack

OpenStack is an open and scalable cloud management platform (CMP) for building public and private clouds. It is a system designed to provide infrastructure as a service (IaaS) on top of a diverse collection of hardware and software infrastructure technologies. OpenStack is free and open-source software released under the terms of the Apache license and managed by the OpenStack Foundation, a nonprofit corporate entity established to promote both the software and its community.

OpenStack is more than a single piece of software: It is a specification that defines a set of functions required to implement cloud solutions. Components that implement these functions communicate via well-defined interfaces. This is more than reference architecture; components of this stack are implemented as various subordinate elements or services, each known by its project code name:

- **Keystone** – Identity service
- **Horizon** – Web GUI
- **Nova** – Compute service
- **Glance** – Image service
- **Neutron** – Network services (formerly called Quantum)
- **Cinder** – Block storage service
- **Swift** – Object storage service

Figure 1 illustrates how these components fit together under the Horizon management solution and are bound together using Keystone for identity management.

---

**Figure 1. OpenStack Components**
These are the actual components that can be downloaded, installed, and configured to manage a cloud. OpenStack is “open” in two ways: First, this software is open source and freely available. Second, and more important for our purposes here, the architecture is pluggable—any of its elements can be replaced by another that implements the specified interfaces and provides the required functionality. In other words, it is a framework with open APIs.

1.3 Using OpenStack with vSphere

vSphere has a long history of being a stable and resilient platform that offers many benefits to host cloud infrastructures. As an enterprise-class hypervisor with production-level features and support, vSphere is an excellent solution for enhancing OpenStack. Many vSphere features facilitate the implementation of OpenStack by simplifying configuration and reducing the number of steps required to provide resources.

vSphere platform capabilities are exposed to OpenStack using drivers that map OpenStack requests into equivalents that VMware solutions can interpret. VMware provides these drivers to the OpenStack community free of charge. Drivers for Cinder and Nova, illustrated in Figure 2, are examples. Significant effort is being applied to the creation of additional drivers, such as one that leverages VMware NSX™ to provide advanced networking functionality via Neutron.

![OpenStack and VMware vSphere](image-url)
1.3.1 OpenStack and vSphere: Conceptual Analogies

When looking at using these technologies together, it is helpful to understand how each of them works. Currently, many administrators come either from the cloud or the infrastructure administration background, and few have experience on both sides of a CMP. The following mapping of OpenStack and vSphere constructs and actions aims to facilitate greater understanding. These are approximate comparisons rather than absolute equivalents.

<table>
<thead>
<tr>
<th>OPENSTACK</th>
<th>VMWARE VSPHERE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flavor + Image</td>
<td>Template</td>
</tr>
<tr>
<td>Instance</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>Volume</td>
<td>VMDK</td>
</tr>
</tbody>
</table>

Table 1. Relating OpenStack and vSphere Constructs

There is a subtle difference between the ways that OpenStack and vSphere deal with “master images.” In vSphere, a template is a full virtual machine specification, including the OS image. In OpenStack, there is a degree of separation: The flavor is the virtual machine specification; the image is the OS.

There is a fairly similar set of actions that can be performed on these objects, although there are again some slight differences in terminology and net effect. OpenStack combines some vSphere actions into a single action. It is important to understand these relationships to prevent unexpected loss or orphaned data.

<table>
<thead>
<tr>
<th>OPENSTACK</th>
<th>VMWARE VSPHERE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Launch Instance</td>
<td>Deploy and power on a virtual machine</td>
</tr>
<tr>
<td>Terminate Instance</td>
<td>Stop and delete virtual machine from disk</td>
</tr>
<tr>
<td>Create Volume and Attach to Instance</td>
<td>Add new VMDK to virtual machine</td>
</tr>
<tr>
<td>Attach VMDK to Instance</td>
<td>Add existing VMDK to virtual machine</td>
</tr>
<tr>
<td>Detach Volume</td>
<td>Remove VMDK from virtual machine without deleting VMDK from datastore</td>
</tr>
<tr>
<td>Delete Volume</td>
<td>Remove VMDK from datastore</td>
</tr>
</tbody>
</table>

Table 2. Relating OpenStack and vSphere Constructs

Typically, the roles of OpenStack administrator and infrastructure administrator are separate, so the preceding information will help those individuals find common ground when discussing the environment as a whole. Such an environment, with many parts working together, can be very complicated and more susceptible to errors. As such, anything that can be done to reduce the management burden is generally a welcome addition.

1.4 The VMware OpenStack Virtual Appliance

A production implementation of OpenStack requires several base components that must be implemented and connected to one another prior to attaching the hypervisor and deploying the first workload.

The VMware OpenStack Virtual Appliance (VOVA) simplifies OpenStack deployment into a vSphere environment for test, proof-of-concept, and education purposes. VOVA has been configured to run all required OpenStack services (Nova, Glance, Cinder, Keystone, and Horizon) in a single Ubuntu Linux appliance.
NOTE: VOVA is not a commercial product and does not come with any kind of warranty or support. It is a learning tool released by VMware engineers to make it easier for VMware customers to try out OpenStack with vSphere. VOVA should not be used for or along with any production workloads.

Interested persons who do not have a lab or other nonproduction environment in which to test can sign up for free access to the VMware Hands-on Labs to experience OpenStack in a test environment: http://vmware.com/go/openstacklab.

2. Requirements

To use OpenStack with vSphere, there are some infrastructure prerequisites that must be met. If vSphere is now implemented in your environment, these are likely similar to your current practices. The remainder of this document assumes that you are using VOVA to deploy and operate your OpenStack environment.

To keep the configuration of VOVA simple, we have made a few assumptions about your environment. For a successful deployment and optimal experience, ensure that your environment is properly prepared prior to downloading and deploying the VOVA Open Virtualization Format (OVF). This document outlines the requirements to implement VOVA for proof-of-concept and education purposes. Additional consideration should be given for production implementations of OpenStack in your environment.

2.1 vSphere Requirements

vSphere compute capacity is exposed to OpenStack via the vSphere cluster object. Using this presentation, OpenStack can model the entire cluster as a single pool of capacity. This can greatly simplify an OpenStack configuration because a single vSphere cluster object can represent the compute capacity of as many as 32 hosts.

2.1.1 vSphere Inventory: Single Datacenter
The current version of VOVA should be used only with a VMware vCenter Server™ system that has a single data center configured. This is a temporary limitation, but for now, VOVA recognizes only one.

2.1.2 Cluster: Automated VMware vSphere Storage DRS
To take advantage of provisioning from OpenStack, a vSphere cluster must be created and VMware vSphere Storage DRS™ with auto placement must be enabled; this means partially automated vSphere Storage DRS at the least. Because OpenStack relates to the vSphere infrastructure from the cluster level, it relies on vSphere Storage DRS to spread virtual machines across all capacity available in the cluster. Because we are discussing cloud and IaaS here, we recommend implementing fully automated vSphere Storage DRS so the balance within the cluster will be maintained automatically as instances are deployed and destroyed.

This implementation provides a great deal of flexibility. Because OpenStack does not detect which specific host is currently running each virtual machine, the virtual machine can be transparently moved to any host in the cluster without impacting its manageability. From an infrastructure perspective, this means that VMware vSphere vMotion® can be used to manually relocate virtual machines, vSphere Storage DRS can be invoked to balance the workload after deployment, and maintenance mode can still be used for host-level management—all without requiring changes to the OpenStack configuration.

2.1.3 Storage: Shared
To support the required cluster configuration, shared storage must be available to all hosts in a vSphere cluster presented to OpenStack. The storage capacity of the shared datastore(s) will be used both for the primary disk of the instances created via OpenStack Nova and for persistent block storage volumes created via OpenStack Cinder. We will discuss this further in the “Managing Storage” section.

For proof-of-concept and education purposes, we recommend use of a single datastore for hosting OpenStack workloads.
2.1.4 Networking

The OpenStack Nova compute service provides basic network connectivity capabilities. In a vSphere environment, Nova basic connectivity leverages a vSphere standard switch or VMware vSphere Distributed Switch™. The more advanced Neutron networking service is available in full OpenStack deployments, but its implementation and integration with VMware NSX is beyond the scope of this document.

2.1.4.1 Port Groups and VLANs

Each VMware ESXi™ host must have a virtual machine port group that provides network connectivity between VOVA and vCenter Server and to whichever client you will be using to reach the OpenStack Web GUI. This is your “management” or “public” interface; you can choose any name for it. For simplicity, we use the default name of *Virtual Machine Network* in our environment.

![Port Groups Diagram](image)

You must create a separate vSphere port group for connectivity between all virtual machines provisioned by OpenStack and between those virtual machines and the VOVA appliance. This is your “private” network and will be attached to the *br100* interface of the VOVA virtual machine.

In our environment, we have created a port group called *br100* for this purpose. If standard virtual switches are used, this port group must be created on each host in the cluster. If a vSphere Distributed Switch or a single ESXi host is used, it can be created once. This port group should be deployed to an isolated VLAN because VOVA runs a DHCP server on this network.

2.1.4.2 ESXi Firewall

OpenStack uses virtual network computing (VNC) to provide access to the consoles of instances it deploys and manages. To get VNC working with the OpenStack Web GUI, the ESXi firewall on each ESXi host must allow the appropriate VNC ports (5900–6000/tcp) through.
2.1.4.3 DHCP Server
The VOVA runs a DHCP server on its second interface—the one that has the br100 designation and is connected to the “private” network. It is best to put this on a dedicated, isolated VLAN to prevent conflicts with an existing DHCP server or “leakage” of DHCP onto other networks in your environment. In a single host deployment, this can be accomplished by creating a port group that has no external uplinks assigned.

For more information on any of these requirements, see VMware OpenStack Virtual Appliance – Deployment Instructions, which is available on the VMware OpenStack Community Web page: http://www.vmware.com/go/openstack.

3. Installation
VMware OpenStack Virtual Appliance – Deployment Instructions provides the latest detailed information for deploying and configuring VOVA in your environment; this section provides a high-level overview of that process.

3.1 Importing VOVA
VOVA is packaged as an OVF that must be imported to your vSphere environment to be used. The OVF should be deployed into the same cluster that will be used for virtual machines provisioned by OpenStack, but only after the cluster has been configured according to the guidelines in the “Requirements” section. Cluster preparation details and import process are discussed in detail in VMware OpenStack Virtual Appliance – Deployment Instructions.

3.2 Configuring VOVA
During the initial OVF deployment, you are required to map the network interfaces and enter various configuration parameters for your environment. Enter them correctly the first time because these values populate various configuration files within the appliance at first boot.

From a network mapping perspective, ensure that you map the “Management Network” of VOVA to the Virtual Machine Network port group—or whatever you called the “public” network in your lab—and the br100 network in VOVA to the port group that connects to your “private” network.

In addition to the interface mapping, the following information will be required to successfully deploy VOVA:

1. IP information for VOVA
   a. Static IP
   b. Netmask
   c. Gateway
   d. Domain Name System (DNS) server(s)
2. vCenter Server host name or IP address
3. vCenter Server port number (default of 443)
4. vCenter Server username
5. vCenter Server password
6. Data center name
7. Cluster name
3.3 Starting VOVA

After the OVF has been deployed, the only remaining step is to power on the virtual machine and then wait for it to boot completely, at which time the blue appliance information page will appear:

![VOVA Console Screen](image)

You now have a proof-of-concept OpenStack environment running. Proceed to the next section to discover what it can do.

3.4 Configuring the VMware vSphere Web Client Plug-in for OpenStack

The VMware vSphere Web Client plug-in for OpenStack enables vSphere Web Client to map OpenStack virtual machine instance properties as they are defined in the OpenStack dashboard portal. This enables vSphere administrators to successfully account for the OpenStack instances within the vSphere infrastructure.

To configure the plug-in, the VOVA appliance must be successfully deployed with a valid IP address as is shown in Figure 4. The plug-in then is ready to be configured and used with vCenter Server and vSphere Web Client.

To configure the plug-in, log in with administrative credentials to vCenter Server using vSphere Web Client. Navigate to the Administrator Option screen and select the new OpenStack option available on the screen as is shown in Figure 5.

![Administrator Option Screen](image)

Click the + icon in the OpenStack Configuration window to register the vCenter Server and Keystone endpoints.
NOTE: The endpoints are the systems the plug-in will be interacting with to retrieve the OpenStack-related information for the vSphere inventory.

Add and configure the vCenter Server endpoint by selecting it and then providing the vCenter Server URL as https://<vCenter Server>/sdk. Then provide the credentials for accessing vCenter Server. After clicking Add, the vCenter endpoint appears with a green check icon as is shown in Figure 6.

![Figure 6. Administrator Option Screen with Active vCenter Endpoint](image)

To avoid configuration errors, ensure that the vCenter Server and VOVA times are synchronized. A time difference of greater than 10 minutes between them will result in endpoint communication failures.

Add and configure the Keystone endpoint by selecting the Keystone Endpoint Type as KEYPSTONE. Provide the URL as http://<VOVA IP>:5000/v2.0. Then provide the VOVA administration credentials to access the OpenStack inventory across all the tenants in this case (admin/vmware). After clicking Add, the Keystone endpoint appears with a green check icon as is shown in Figure 7.

![Figure 7. Administrator Option Screen with Active Keystone Endpoint](image)

After the vCenter Server and Keystone endpoints have been successfully added and configured, a green check icon appears at the top of the OpenStack Configuration screen as is shown in Figure 8. The plug-in configuration is now completed.
After the endpoints have been configured, searchable tags are created for virtual machine instances on vSphere whenever the OpenStack portal or CLI is used to deploy those instances. For example, when an instance is launched and named “RawlinsonVM” in OpenStack, the name of the instance becomes a searchable item in the vSphere inventory as is shown in Figure 9.

Using vSphere Web Client, search operations now can be performed based on virtual machine instance names, such as with the previously cited example.

The plug-in also adds a new portlet, OpenStack VM, to the Virtual Machine Summary screen as is shown in Figure 10. When navigating the Virtual Machine Summary tab, the screen displays the OpenStack virtual machine portlet, which contains the attributes related to the OpenStack virtual machine instance as they are available in the OpenStack portal.
4. Managing OpenStack with the Horizon Web Interface

There are two options for managing your new environment: Web-based and command-line interface (CLI). For most first-time users, the easiest way to get started is to use the Web-based GUI. This GUI is known in an OpenStack context as Horizon, which is not related to VMware® Horizon™.

4.1 Logging In – Web

You can manage your new environment by pointing your Web browser to the OpenStack Dashboard referred to on the information screen displayed on the VOVA console: http://management_IP_you_entered/.

The Horizon Web GUI on VOVA utilizes an initial username “demo” with the password “vmware.”

4.2 Logging In – SSH/CLI

If you are a command-line user with previous OpenStack experience, you might prefer to use the command-line tools: Open your favorite SSH client and point it at the management IP of the VOVA. Log in as “root” with the password “vmware.”

For more information, including examples of how to manage OpenStack using the command line, see VMware OpenStack Virtual Appliance – Deployment Instructions and OpenStack documentation. The remainder of this document will focus on using Horizon for management.

4.3 Flavor of the Day

In OpenStack, a “flavor” is a virtual hardware template, defining sizes for RAM, disk, number of cores, and so on. This is roughly equivalent to a “template” in a vSphere context, except that a “flavor” does not actually contain the OS image. In an OpenStack context, the OS binaries are stored in a separate construct called an “image.” VOVA comes preloaded with a single image for Debian Linux and a few flavors that can be used with that image.

![OpenStack Flavors](image)

Figure 11. OpenStack Flavors
The Debian image provides a functional example that consumes relatively few resources and boots quickly. Adding your own images and flavors is beyond the scope of this document.

4.4 Launching an Instance

An “instance” in OpenStack is roughly analogous to a “virtual machine” in vSphere. There are some differences, but for our purposes they are the same. When an instance is launched in OpenStack, a vSphere virtual machine is created and powered on.

To manage OpenStack instances in the Horizon interface, ensure that the Project tab is selected and navigate to the Instances tab on the left side of the page.

![OpenStack Instances](image)

Figure 12. OpenStack Instances

When you launch an instance, it will be built according to the specifications defined by the flavor you select and will be preloaded with the Debian Linux OS. To deploy your first instance, ensure that the Project tab is selected on the left side of the screen, and then click Instances. Click the Launch Instance button on the upper right side of the screen.

Select Debian from the Image drop-down menu, enter a name for the instance in the Launch Instance dialog, and click Launch.
Getting Started with OpenStack and VMware vSphere

Your instance will enter the “Spawning” state while it is created and booted, at which point it will enter the “Running” state.

4.5 Accessing the Console for an Instance

After your instance is up and running, you can access the console directly from the Web browser by clicking More in the instance’s Actions column and choosing Console.
Getting Started with OpenStack and VMware vSphere

Figure 14. Accessing the Console for a Running Instance

This will make a VNC connection to the running instance and present the console inside the Web browser.

Figure 15. The VNC Console for an Instance
4.6 Managing Storage

Although an instance is excellent for providing access to compute (memory, CPU) and networking, an instance in OpenStack is an ephemeral entity: *When the instance is terminated, it is removed from the infrastructure.* This means that any data created inside the instance is also removed. From a vSphere perspective, the virtual machine is deleted from the disk; however, leveraging Cinder for persistent block storage can preserve the work that an instance has performed.

4.6.1 Adding Persistent Storage to an Instance

To get started, click **Volumes** in the Horizon interface.

![Creating a Persistent Volume](image)

Next, click **Create Volume** in the upper right to begin creating a new volume. Enter a name and a size for the new volume, and click the other **Create Volume** button.

After the volume has been created, it can be attached to an instance by clicking **Edit Attachments** in the volume’s **Actions** column.
Select the instance that will receive the volume and enter an appropriate **Device Name**. The volume will be made available at this location in the instance’s device tree.

Persistent storage is used within an instance, as with any other block device; it must be discovered, partitioned, formatted, and mounted.

After the volume has been attached, perform volume discovery in the Debian Linux instance provided with VOVA, by logging in with the password “vmware” as the root user and running the `/opt/scan.sh` command. After this has been completed, the device will be available in the location you specified.

For quick reference, the following high-level steps are required to complete this process:

1. Use the Horizon interface:
   a. Navigate to the **Volumes** tab.
   b. Click **Create Volume**.
   c. Provide a volume name and size; click **Create Volume**.
   d. Click **Edit Attachments** for the volume.
   e. Select the **Instance** and provide a device path; click **Attach Volume**.

2. Log in to the instance:
   a. Scan for new volumes.
   b. Partition the volume.
   c. Format the volume.
   d. Mount the volume.
4.6.2 Removing Persistent Storage from an Instance
When an instance no longer is needed and its storage must be preserved, the volume can be removed by reversing the process: dismount the file system within the instance and disconnect the volume by using the Horizon interface. The volume still exists in the Volumes list even after it has been detached from the instance.

For quick reference, the following high-level steps are required to complete this process:

1. Log in to the instance:
   a. Ensure that the volume is not being used.
   b. Dismount the file system.

2. Use the Horizon interface:
   a. Navigate to the volume.
   b. Click Edit Attachments for the volume.
   c. Click Detach Volume: acknowledge the warning.

4.6.3 Adding the Persistent Storage to Another Instance
When another instance has been launched and requires access to this volume, the steps to attach the volume to the instance are similar to the initial volume creation, except that the volume should not be partitioned or formatted after being attached to the instance; that would result in loss of any data on the volume.

For quick reference, the following high-level steps are required to complete this process:

1. Use the Horizon interface:
   a. Navigate to the volume.
   b. Click Edit Attachments.
   c. Select the Instance and provide a device path; click Attach Volume.

2. Log in to the instance:
   a. Scan for new volumes.
   b. Mount the volume.
4.7 Network

The current version of VOVA uses the basic networking provided by Nova. Nova provides a very simple and flat IP address space similar to that provided by a VMware virtual switch.

Using this simple mechanism, advanced features such as security groups are not supported. However, that functionality will be enabled soon via the enhanced Neutron networking and integration with VMware NSX. As with the current Nova and Cinder support, Neutron integration will be enabled via a driver. This driver will communicate directly with the VMware NSX controller rather than vCenter Server, as is shown in Figure 18.

![OpenStack Neutron and VMware NSX](image-url)
Summary

In this document, we have shown how OpenStack can be used as a cloud management platform on top of a vSphere compute and storage infrastructure. Using the appropriate drivers, OpenStack enables self-service consumption of compute and storage in the form of Nova compute instances and Cinder block storage volumes with VMware vSphere as the provider.

Acknowledgments

We would like to thank Dan Florea and Dan Wendlandt of the VMware OpenStack team, whose deep knowledge and understanding of OpenStack was leveraged throughout this paper. I would also like to thank Charu Chaubal, group manager of the Storage and Availability Technical Marketing team, for reviewing this paper.

About the Authors

Rawlinson Rivera is a senior architect in the Cloud Infrastructure Technical Marketing group at VMware. His focus is on storage virtualization, software-defined storage technologies, and integration aspects of VMware products and solutions with the OpenStack framework.

Previously he was an architect in the VMware Cloud Infrastructure and Management Professional Services organization, focused on vSphere and cloud enterprise architectures for VMware Fortune 100 and 500 customers.

Rawlinson is among the first VMware Certified Design Experts (VCDX#86) and is the author of multiple books based on VMware and other technologies.

Follow Rawlinson’s blogs:

• http://blogs.vmware.com/vsphere/storage
• http://www.punchingclouds.com

Follow Rawlinson on Twitter:

• @PunchingClouds

Doug Baer is an infrastructure architect in the Cloud Infrastructure Technical Marketing group at VMware and splits his time between working on VMware Hands-on Labs and other projects. Previously, he was an infrastructure architect and consultant with a VMware partner, where he focused on solving customer challenges and implementing a variety of server, storage, and VMware solutions. Doug is a VCDX (#19) and an HP Master ASE – data center and cloud architect.

Follow Doug’s blogs:

• http://blogs.vmware.com/hol
• http://blogs.vmware.com/vsphere

Follow Doug on Twitter:

• @dobaer